A Multiscale Discrete Model Integration Strategy For Systems Biology Implemented in a Grid-Enabled Software Platform: An Example Application from Cancer Systems Modelling

Submitted in fulfilment of the degree of Doctor of Philosophy at the
Department of Oncology (Royal Free Campus), University College London

Supervisor: Dr Sylvia Nagl

30 September 2007

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

I, Manish Patel, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Manish Patel

Date:

ABSTRACT

Model integration – the process by which different modelling efforts can be brought together to simulate the target system – is a core technology in the field of Systems Biology. In the work presented here model integration was addressed directly taking cancer systems as an example. An in-depth literature review was carried out to survey the model forms and types currently being utilised. This was used to formalise the main challenges that model integration poses, namely that of paradigm (the formalism on which a model is based), focus (the real-world system the model represents) and scale.

A two-tier model integration strategy, including a knowledge-driven approach to address model semantics, was developed to tackle these challenges. In the first step a novel description of models at the level of behaviour, rather than the precise mathematical or computational basis of the model, is developed by distilling a set of abstract classes and properties. These can accurately describe model behaviour and hence describe focus in a way that can be integrated with behavioural descriptions of other models. In the second step this behaviour is decomposed into an agent-based system by translating the models into local interaction rules. These rules must be enriched and the agent model simulated, therefore a Grid-like Java infrastructure was developed and tested on an 18-node Beowulf cluster. The two-tier approach was tested on this software by taking 4 different models, each exhibiting complexities and submodels, that were successfully integrated and simulated together. The results showed all of the main challenges could be overcome given the correct conditions for rule enrichment, in this case implemented as a genetic algorithm that operated on rule components.

This research represents a key breakthrough for cancer systems research. The two-tier approach could provide the tools necessary to understand tumour behavioural complexity and hence provide a means to combat the disease.

ACKNOWLEDGEMENTS

First and foremost my deepest thanks to Sylvia who has not only paid more attention to my work than I could possibly hope for from a supervisor she has also been a good friend. Equally, my most sincere thanks to my family – my mums and dads especially – and friends who, despite my absence from their lives due to my work, have stood by me without question. My thanks must also be extended to everybody at the department without whom I could not possibly have completed this project. Additionally it is the studentship offer made by the Medical Research Council that enabled our group to carry forward this research and my thanks is extended not only to the MRC but also those who were involved in securing that financial support.

And Tanvi, who has supported me emotionally throughout – without her I would not have even made my final decision to undertake and complete this work in the first place.

There is not enough space in this entire thesis to thank all the people, who have one way or the other been connected to my work, that deserve my most heartfelt gratitude and any effort made in this vain will only ever be incomplete – to those who I have left out I apologise but rest assured that I will not forget what you have done for me.

For that Person who has never left my side

jnanena tu tad ajnanam yesam nasitam atmanah tesam aditya-vaj jnanam prakasayati tat param

Just as the sun lights up the day, when one is enlightened with the knowledge by which nescience is destroyed, then his knowledge reveals everything.

Bhagavad Gita 5:16

CONTENTS

1: INTRODUCTION

Biology is shifting into an era of research where the use of mathematical and computational methods is becoming an integral part of the investigative cycle. Whilst traditionally engineering and computer science-oriented fields have guided biology in this respect, keeping themselves separate as disciplines, *systems biology* – a growing field in which biology, computer science and engineering meet – is quickly becoming a focal point for complexity research and modelling. This has been driven by technological innovation, e.g. microarrays and high-performance computing, enormous endeavours in scientific discovery, such as the genome projects, and the sheer necessity of disease prevention and cure. The fact is that there are no known systems more complex than the human body and other biological systems and therefore the opportunity for research is unique.

Currently at the core of the systems biology effort is the determination of behaviour at all levels of the system, from molecular to whole-organism. The benefits of such a wide understanding, once achieved, will be immense – not only will it yield greater understanding for the purposes of medicine, there is additionally a reciprocal stream by which disciplines engage ideas from biology, e.g. bio-inspired computing[1].

Modelling plays a fundamental role in this challenge. It is invariably a multifaceted process that starts with conceptualising the target system – often simplifying it and dissecting it into manageable modules – upon which a set of mathematical and computational formalisms is based. It can be said of these formalisms:

- Their structure and convention can be extremely diverse.
    - For example, take as a 'random sample' differential equations and Boolean networks. Not only are these fundamentally different in design, the process of solving and executing them is completely different.

[1] This has been seen before in computer science, for example, where understanding of evolutionary processes has led to a great many number of applications in the form of evolutionary algorithms.

- Their components can become extremely complex.
    - As will be seen, some modelling efforts encapsulate such depth of detail as modelling formalisms for complex systems are often difficult to solve without simplification.
- The model and its solver are sometimes difficult to separate.
    - An example of a solver technology is a differential equation solver. Some models have actually been published as complete packages wherein the model and the solver are tightly coupled.
- The semantics of such models are often difficult to conceptualise into a precise computable form.
    - Assumptions, context, data sources, caveats and simplifications are all usually written in free text since there is no standardised way to describe such metadata.
- Biological systems, and hence models, operate on a multitude of scales.

What is the nature of this diversity? How does this diversity relate to the real system? What is the commonality between these diverse formalisms? Can there be any *unity* in this diversity? These are crucial questions that have to be asked if modellers are to make their simulations more representative of the real system. The reason is that each effort focuses on different aspects of the system and they ultimately need to be integrated. Moreover the composition of elements within the same system can be such that they warrant different modelling approaches. Unifying principles must therefore be found in order to harness the full potential of individual modelling efforts, paving the way to a more complete understanding.

The most common name given to this unification is *model integration*. Implementation has taken different forms within the few disciplines in which it has been tackled, however the underlying principle and aim has remained constant: model integration in its most common conceptualisation is a solution by which disparate models are executed in such a way that the resulting model and output(s) are concordant with the original models and represent a model of a larger system of which the individual models are constituent parts.

The difference in implementations is completely dependant on the nature of modelling methodologies within those fields of research. For example, as will be discussed in Chapter 4, the Management and Operations Research sciences have developed a kind of integration strategy that is applicable to the sequential nature of the models found there.

There does not seem to be any universal method by which one can integrate models. The necessity for such a technology is widely recognised, and is equally evident in the field of cancer systems biology. However, its development is somewhat stunted when compared to other areas of research. Primarily the reason for this is the progressive complexity of models and simulations used and hence their integration is not forthcoming.

The recent rapid development of mathematical models describing complex biological processes in cancer at all scales, encouraged by the pressing requirement for improved therapy, makes tumour systems a compelling target for model integration research.

For these reasons this project has aimed to meet this challenge head-on and apply it to cancer systems biology. Firstly an in-depth literature review is given in Chapter 2. This review discusses the current state of the tumour modelling literature, where research is heading and the benefits and shortcomings of current methods. Not only is this a review of representative models and techniques, it serves as the basis on which the aforementioned diversity of models is analysed in more detail. This analysis provides the information needed to investigate the precise challenges that model integration presents.

The types of approaches in any modelling-oriented field are parallel to the nature of the real system itself. This is especially true in tumour systems biology where the mathematical and computational techniques applied to tumour biology are found to be correspondingly intricate to be able to explain its highly non-linear behaviour. Therefore the science of complexity and systems in the context of cancer biology is described in relevant detail in Chapter 3.

Based on the findings of the literature review and insights into complex systems, a two-phase approach is developed, explained in Chapters 4 and 5, and implemented as a parallelisable Java application, described in Chapter 6. The method borrows ideas from complex systems as agent-oriented systems and object-oriented concepts from software engineering to tackle model complexity and semantics. The theory is tested by integrating models from the literature and an analysis of the results is made in Chapter 7. Finally, the theoretical basis of the method is extrapolated to include additional technologies in Chapter 8 to show that not only is it a multi-disciplinary approach, but also to illustrate that it is a well-rounded approach that shows promise for cancer modelling as well as other modelling disciplines.

2: A REVIEW OF MODELLING IN CANCER SYSTEMS BIOLOGY

2.1 Chapter Objectives

This chapter represents a survey of the literature pertaining to mathematical modelling of tumour biology and techniques used to achieve tumour simulations. The purpose of this review is:

- To portray the vast array of techniques currently in use in the field, including mathematical and computational formalisms.
- To develop an appreciation of the enormous complexity of the problem domain, i.e. the tumour system, and hence the parallel complexity of the techniques needed to describe the system.
- To develop a detailed survey of model attributes so that the challenges that model integration poses, as well as possible solutions, can be defined.

The mathematical and computational formalisms of the models reviewed below are described in Appendix A1, applying the novel classification scheme developed in this present work.

This review has been published as a chapter in Nagl (2006), the first full-length text on cancer bioinformatics[1].

2.2 Mathematical Modelling in Tumour Systems Biology

Definition 2.1: Model** and **Simulation

A model can be defined as an abstraction of a real system (Mooney & Swift, 1999). The abstractions of interest here are those that are expressed in some algorithmic form, be it mathematical, computational or logical. "Simulation" and "model" are often used

[1] University College London Cancer Institute:
http://www.ucl.ac.uk/cancer/research-groups/cancer-systems-science/index.htm

interchangeably in the literature, however the term "simulation" can be defined as a model that is executed via a solver over a parametric variable, e.g. time. The terms "model" and "solver" are defined formally in more detail in Chapter 4.

Mathematical techniques in cancer biology have been applied for many years and their use has dramatically increased with the arrival of fast and accessible computation. The literature shows numerous examples of the use of mathematical modelling techniques, many of which have been previously applied to fields such as physics and environmental/meteorological sciences. It is evident that the application of such techniques is now ubiquitous in cancer research reflecting a general trend in almost all areas of biology and biomedicine. These analytical methods have been used to assist understanding of the emergence of complex tumour behaviour, which will be discussed in this chapter.

The types of available models focused on cancer biology range from general models, e.g. tumour growth, angiogenesis and signalling transduction pathway models, to specific models, e.g. behaviour in response to specific stimuli. Models of biological processes are employed for diverse research purposes, notably, inquiries into fundamental tumour dynamics and drug target prediction. Algorithms used by these models range from differential equations, use of fractal theory (Baish & Jain, 2000), stochastic approaches (Wolkenhauer *et al.*, 2004) and more recently artificial intelligence techniques such as clustering and classification, neural networks, application of fuzzy logic (Catto *et al.*, 2003), inductive/stochastic logic programming (Siromoney *et al.*, 2000), Bayesian networks (Jensen, 2001) and many more. Epidemiological models are also abundant in the literature, which are informative for the clinician in terms of modelling prognoses and diagnoses given various biological data[2].

The types of models presented in this report have been broadly categorised into the following classifications: Growth Models, Angiogenesis Models, Treatment Response

[2] This model class includes statistical models, which does not directly tackle systems behaviour and hence are not included in this review.

Models and Dynamic Pathways Models. Of course not all of the models presented here can be strictly classified into these types – in reality all are hybrids to some degree but most comfortably fall into one of the categorisations, as portrayed by Figure 2.1. As this figure suggests, there are virtually no models that incorporate *all* of these aspects to describe tumour behaviour. The following sections will highlight the fact that inclusion of diverse behaviours yield richer and more precise models. Indeed this is a key point in this project – models have traditionally dissected behaviour in such a way that it is often not obvious how to integrate them back together again. The methods described in this project endeavour to remedy that situation.

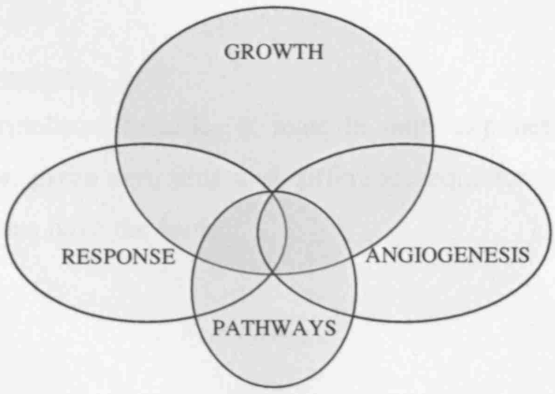


Figure 2.1. Current state of the tumour modelling literature. The majority of models can be classed as growth models; this aspect of cancer has attracted the most attention from modellers since it is one of the main clinically important behavioural features. More mechanistic models include response to treatment, angiogenesis and pathway dynamics. Relatively few models include a mixture of all these aspects.

2.2.1 Growth Models

Growth models are the most abundant types of model for solid tumours in that all other modelling exercises must take into account the intrinsic growth functions that exist within the system.

Chignola *et al.* (2000) point out two important features that characterise the growth of tumours. Firstly, tumours are inherently made up of a heterogeneous population and therefore the growth dynamics can be quite complex. When modelling overall growth one is actually trying to model the growth of several phenotypically different populations.

Secondly, growth is limited and cannot continue indefinitely. The constraint is placed by limitations in resources and space. All models in the literature seem to recognise these characteristics to some degree though it is not always explicitly stated.

The majority of the tumour modelling effort seems to be in the analysis of mechanisms that *control* growth since this is one of the most important factors that affect tumour development and response to treatment (Byrne, 1999). As a reflection of this one can see a rich variety of growth models in the literature dating back to the early 20th century. However, as yet, there is no universally accepted formalism that describes either vascular or avascular tumour growth (Gatenby & Maini, 2003).

2.2.1.1 Generalised Formalisms

The most simplistic formalisms include, at least in part, exponential models within defined bounds based on given restraints and difference equations (Mooney & Swift, 1999). Difference equations have the form:

$$x_t = R \cdot x_{t-1} \qquad \text{Eq. 2.1}$$

where:

x_t is the next discrete state of the system.

x_{t-1} is the current state of the system (previous time index).

R is a constant that denotes the rate of increase or decrease.

More evolved models tend to have complicated functions in place of the parameter R rather than a constant. Collectively this can be termed as the state transition function.

A closed form of Equation 2.1 can be expressed:

$$x_t = x_0 \cdot R^t \qquad \text{Eq. 2.2}$$

which can be approximated to the following exponential when t is large:

$$x_t = x_0 \cdot e^{Rt} \qquad \text{Eq. 2.3}$$

where x_0 is equivalent to the state at the first state/time index (e.g. tumour radius or mass).

One of the first generalised functions used to describe changes in population number, i.e. cell number in tumours, is the Gompertz function, formalised by Benjamin Gompertz in 1825. It was introduced specifically for tumour growth by Winsor (1932). Other well-known functions include the logistic growth function and the von Bertalanffy growth function (these are very similar to the Gompertz function, and have been used to describe tumour growth previously, but will not be discussed any further here since the explanation of the Gompertz model here suitably covers the extent to which these kinds of model can describe tumour dynamics; the reader is directed to Marusic *et al.*, 1994, for a comparative review).

The Gompertz function reads as follows, although many forms of the equation exist:

$$Y = A + Ce^{-e^{-B(t-M)}} \qquad \text{Eq. 2.4}$$

where:

Y represents growth, e.g. tumour volume or radius.

A is the initial value, i.e. the lower asymptote.

C is the maximum asymptote.

B is the growth rate.

t is time.

M is the time of maximum growth rate.

A, C, B and M are all constants.

The Gompertz curve is of course a gross simplification of population increase but is used ubiquitously across many fields, from tumour biology to population studies, ecology and psychology.

The literature suggests that this function can be used successfully to describe tumour growth (Araujo & McElwain, 2004; Adam & Bellomo, 1996; Desoize, 2000, Kunz-Schughart *et al.*, 1998). The Gompertz function yields a curve that exhibits an initial stage of exponential growth, which eventually levels off into a plateau (Figure 2.2). This type of curve suggests an intrinsic increase in growth retardation imposed on a growing number of individuals in the population until equilibrium such that growth and retardation, due to death or stasis, is reached.

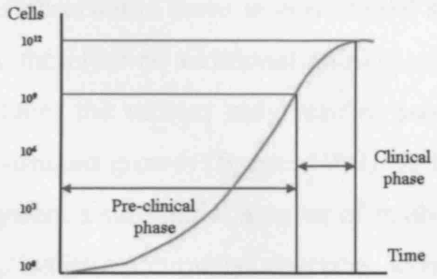


Figure 2.2. A generalised Gompertz curve showing the population increase in relation to time. The graph also shows an estimate of the size at which the tumour is detectable, i.e. a tumour is said to be in the clinical phase when it has actually been detected and is being treated. Image adapted from Hall (2003).

However, the complexities of the tumour system is such that a model needs to be able to capture the rich and highly connected functional modules that the system exhibits, e.g. particular processes orchestrated by cellular pathways, to truly have the capacity to qualitatively and quantitatively mimic the system or command any clinical meaning. Hill (1928) was one of the first to realise this and investigated diffusion of molecules through tissue and its responsibility for tumour behaviour, hence creating one of the first algorithms for growth that integrated tumour biology into mathematics. The following decades saw the emergence of growth models that focused on the actual growth dynamics rather than deeper, analytical algorithms (Araujo & McElwain, 2004).

However not many of these algorithms at the time could actually be validated against the real biological system due to the lack of an experimental tumour system. This changed with the introduction of *in vitro* multicellular tumour spheroids.

2.2.1.2 Multicellular Tumour Spheroids and Tumour Cords

In vitro multicellular tumour spheroids (MTS) were introduced by a number of groups, the most cited being Sutherland & Durand (1971), as model *experimental* systems and a real alternative to the somewhat limited monolayer techniques for the exploration of tumour behaviour (Bates *et al.*, 2000). They have been successfully applied to many areas of the field including therapy resistance, drug penetration, invasion and tumour cell metabolism (Kunz-Schughart *et al.*, 1998, Desoize, 2000). The familiar morphology of a necrotic centre surrounded by a layer of hypoxic tissue, which in turn is surrounded by a layer of normoxic tissue is observed in these *in vitro* model systems (Figure 2.3a). Since there is no vasculature and therefore no additional nutrient supply (apart from that which is available from the surface) the tumour only reaches some small maximum radius, which is termed diffusion-limited growth (Byrne, 1999). In the wake of the accelerated use of this experimental system a substantial number of mathematical models for tumour growth were developed, which accurately describe avascular growth retardation, especially that of spheroid growth.

However, there are some critical issues that must be addressed before accepting that MTS effectively reproduce *in vivo* processes. The first objection that can be raised is the fact that many primary tumours are rarely discovered before neovascularisation has occurred and therefore the clinical significance of a spheroid mathematical model is automatically made immaterial (Gatenby & Maini, 2003). Once a tumour begins to exhibit neovascularisation the entire morphology and behaviour of the tumour system changes. The supplementation of nutrients from blood vessels enables cells that were otherwise quiescent to re-establish their mitotic paths (Mantzaris *et al.*, 2004) unless other local inhibitory factors exist, e.g. cellular overcrowding (Chignola *et al.*, 2000).

It is also apparent that all too often the models are again simplified even more by assuming the boundaries between the morphological layers are crisp, discrete and symmetrical (Sherrat & Chaplain, 2001). The genetic hetero/homogeneity of the MTS has also been brought into question (Deisboeck *et al.*, 2001), i.e. the symmetry of MTS would seem to suggest genetic homogeneity in the cell population whereas it has been shown that this is rarely the case in tumours. However most experts agree that the microenvironment itself, given the context, accurately mimics the *in vivo* system although some biological complexity is lost, e.g. MTS generally exhibit stable chromosome number (Kunz-Schughart *et al.*, 1998, Desoize, 2000).

These limitations considered, one must wonder how much analytical insight, pertinent to clinical phase tumours, models of avascular growth can actually offer. Even so the vast majority of growth models in the literature are spheroid models (Araujo & McElwain, 2004). At the very least, the formulation and analysis of these models, and subsequent refinement, can form a basis for discovery of the factors involved in early tumour growth. It can therefore be seen as a valuable starting point in itself, even in the light of the known shortcomings of this approach.

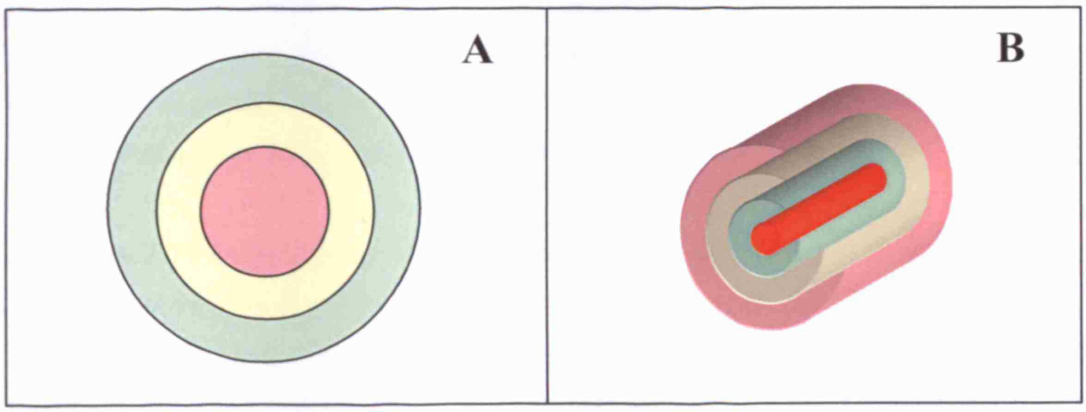


Figure 2.3. **A**. Idealised representation of a MTS; normoxic cells at the periphery (green), hypoxic layer (yellow) and necrotic core (light red). Maximum radius ~1-3mm (Mantzaris *et al.*, 2004). **B**. Idealised representation of tumour cord: vascular centre (dark red) surrounded by normoxic layer (green), hypoxic layer (yellow) and necrotic layer (light red). Maximum radius, including vessel, is approximately 60-140μm (Scalerandi *et al.*, 2003).

As can be seen from Figure 2.3b, the tumour cord is the morphological opposite of the MTS, having the nutrient supply in the middle of the tumour and the layers of cell state types (normoxic, hypoxic, necrotic) surrounding it. This configuration can be thought of as a micro-system of an *in vivo* vascularised tumour. The vascularised tumour in turn can be thought of as a collection of cords. Vascular-driven growth is discussed shortly.

2.2.1.3 Avascular Growth Modelling Formalisms

Assumptions for preliminary modelling of avascular tumour growth almost invariably include a genetically homogenous cell population for each layer, spatial homogeneity and a whole-structure radial symmetry (Byrne, 1999, Friedman, 2004). Many of the models in the literature stop short of vascularisation but there are some groups who have extended their avascular models into the vascular stage. This type of modelling will be discussed in §2.2.2.

It has already been stated that the Gompertz equation forms the basis of many models, and indeed also forms the validation of many models, which will be shown later. The Gompertz function itself, however, is of course purely phenomenological and does not offer any insight to the mechanisms that actually contribute to growth. Chignola *et al.* (2000) take the formalism one step further by formulating a stochastic Gompertz-like growth equation that takes into account the variability in growth dynamics of a heterogeneous population. Even so the model does not give real insight into molecular or cellular mechanisms nor does it shed any light on probably morphology.

Definition 2.2 Phenomenological** and **Mechanistic Models

Phenomenological models provide relatively simple constructs that can mimic a system by means of considering (usually macroscopic) observables only. Conversely mechanistic models in this thesis are defined as those that take into account finer-grained behaviours and components of the system.

Ordinary differential equations (ODE) and partial differential equations (PDE) are by far the most popular mathematical techniques to describe avascular, and indeed vascular,

tumour growth, as well as systems biology[3] in general (Araujo & McElwain, 2004, Byrne, 1999, Friedman, 2004). The most basic equations define tumour growth in terms of available nutrients and/or oxygen supply. Therefore Friedman (2004) states concentration gradients to be described as a PDE due to Fick's 2nd Law:

$$\varepsilon_0 \frac{\partial c}{\partial t} = \nabla^2 c - \lambda c \qquad \lambda > 0 \qquad \text{Eq. 2.5}$$

$$\varepsilon_0 = \frac{T_{diffusion}}{T_{growth}} \qquad \text{Eq. 2.6}$$

where:

ε_0 is a small positive coefficient (e.g. 1 minute/1 day)

$T_{Diffusion}$ is the diffusion time-scale

T_{Growth} is the tumour growth time-scale

c is concentration

λ is a molecular, e.g. nutrient, degradation/uptake coefficient (or rate)

∇ is the gradient operator (Laplacian) given by Equation 2.7

$$c \cdot \left(\frac{\partial}{\partial x} + \frac{\partial}{\partial y} + \frac{\partial}{\partial z} \right) \qquad \text{Eq. 2.7}$$

where x, y, z are Cartesian directions.

Dasu *et al.* (2003) make extensive use of diffusion equations to model oxygen gradients, incorporating static vascular structure (fixed nutrient source), and subsequent effects on tumour growth.

[3] The field of systems biology is discussed in more detail in Chapter 3. For now, systems biology can be defined as a field of study that encompasses both system complexity and biology.

Byrne (1999) describes basic avascular (symmetrical) tumour growth in the form of an ODE:

$$\frac{1}{3}\frac{d}{dt}(R^3) = R^2\frac{dR}{dt} = \int_0^R S(c)H(r - R_N)r^2dr - \int_0^R N(c)H(R_N - r)r^2dr$$

Eq. 2.8

where:

$\frac{1}{3}\frac{d}{dt}(R^3)$ is rate of change of tumour volume over time.

$\int_0^R S(c)H(r - R_N)r^2dr$ is net rate of cell proliferation ($S_{(c)}$).

$\int_0^R N(c)H(R_N - r)r^2dr$ is rate of necrotic cell death ($N_{(c)}$).

R_N is necrotic radius.

R/r is overall radius and hypoxic radius respectively.

H is the Heaviside step function.

Byrne (1999) goes on to describe how one can extend Equation 2.8 into further models that account for asymmetry by vascularisation and multiple cell populations. A more heterogeneous approach can be taken by reapplying the above equations with different coefficient terms to reflect different growth characteristics of multiple cells types. Friedman (2004) also shows how these equations can be extended to multiple cell populations but simultaneously recognises that mixed population models, where different cell-type populations are continuously present everywhere in the tumour, are more realistic than the normal segregated models where cell-type populations are assumed to have discrete boundaries. Sherrat & Chaplain (2001) tackle the problem of discrete interfaces of cell populations by incorporating ODE models that describe densities of cell populations with cellular movement along pressure gradients.

The differential equation approaches described above suffer from some fundamental limitations. It is generally agreed that differential equations by themselves do not capture

enough resolution in terms of spatiotemporal behaviour of the system. At the very least, to get more information out of the models a great number of equations must be generated and solved (Succi *et al.*, 2002), which might prove impossible in some applications, as some physical data needed for parameterisation might not be available. Modelling 3D heterogeneity becomes a computationally challenging task. Another problem, which is apparent from the equations stated above, is that the use of differential equations automatically assumes that the current state of the system is a consequence of the previous *global* state of the system. In reality, a single cell will behave according to its immediate locality, not according to the state of the tumour as a whole, and local environments are differing. Indeed, as Gatenby & Maini (2003) point out, "too often we are content with work that is entirely phenomenological – 'curve-fitting' data – without developing mechanistic models that provide real insights into the critical parameters that control system dynamics". However there are indications of a paradigm shift in the literature with regard to mathematical modelling of both avascular and vascular tumours.

The literature appears to be drifting away from the traditional compartmental, ODE and PDE approaches to mathematically simpler methods such as cellular automata (CA) and agent-based modelling (ABM). This could be due to the increase in access and popularity of powerful computational techniques and software as well as the surge of interest in biological systems from the computer science community.

2.2.1.4 Non-traditional Avascular Growth Models

The use of CA models has become more prominent in recent years. One of the earliest CA[4] models for tumour growth was introduced by Duchting & Vogelsanger (1981). Qi *et al.* (1993) took its use further, formulating a stochastic two-dimensional CA that took into account proliferation, nutrient supply, mechanical pressures (and thus motion of cells within the tumour) and immune surveillance rules. The model quite plainly shows how simple rule definitions in CA can yield complex behaviour and, in this case, qualitatively

[4] Note that there are many different types of CA. Some implementations are very much traditional but may differ in some fundamental ways, e.g. a CA is required to have an identical rule set that does not change. However some implementations might break this rule and so, though thematically still a CA, cannot be strictly classed as one. The work by Stamatakos' group is an example of a CA-like implementation (see main text). CA are discussed in more detail in Chapter 4.

accurate simulation results. The discrete grid cells react according to a neighbourhood of four other discrete cells (von Neumann neighbourhood) and can be occupied simultaneously by cancer/normal cells and macrophage cells. The discrete cells of the CA therefore represent populations of biological cells, or densities. Macrophages can engulf cancer cells (if they are coexisting), which is defined by a fixed probability, and/or cancer cells can divide. Cancer cells are also allowed to invade neighbouring discrete cells occupied by normal tissue. Both proliferation and invasion have a space restriction and global growth, i.e. overall radius, is defined as a function of the internal pressure of the tumour (which itself is a function of cellular densities and proliferation). These simple rules are formalised into the CA and it was found that the resulting growth followed a Gompertz-like growth law. Scalerandi and co-workers describe a discretised two-dimensional approach to modelling cord growth dynamics considering competition for nutrients and conservation of energy (Scalerandi *et al.*, 2002, 2003). They highlight the use of locality-interaction modelling where cells react to their local environment and consequently demonstrate good agreement with experimental data. This is in sharp contrast with the phenomenological models mentioned earlier where the focus is on the global state of the system. The local interaction approach has the added benefit of incorporating heterogeneity in a way that is clearly not possible with phenomenological or differential equation based techniques.

Stamatakos *et al.* (1998, 2001), provide a prime example of how one can model a tumour without using the complex mathematics of differential equations but at the same time capture rich spatiotemporal qualities of avascular tumour growth. As can be seen from Figure 2.4 the structure of a tumour bears remarkable resemblance to the spheroid structure. Stamatakos *et al.* use simple rules and phase transitions of cellular cytokenetics to simulate avascular tumour growth. A discrete space/time formalisation was employed with a rule base very much akin to the CA formalism. The virtual cells are made to run simplified cell cycle phases, which is reminiscent of the multiphase approach adopted by Please *et al.* (1998). More recently the same group developed integration between an improved version of the model, including the oxygen enhancement ratio (OER) and linear quadratic (LQ) models (Equations 2.10-2.12) in the rules for cells, and an

algorithm that relates neovascularisation information from brain scan images of glastiomas (Antipas *et al.*, 2004). Dionysiou & Stamatakos (2006) further incorporate p53 status to the radiosensitivity model thereby yielding a simulation that can predict the effects of different radiotherapy regimens.

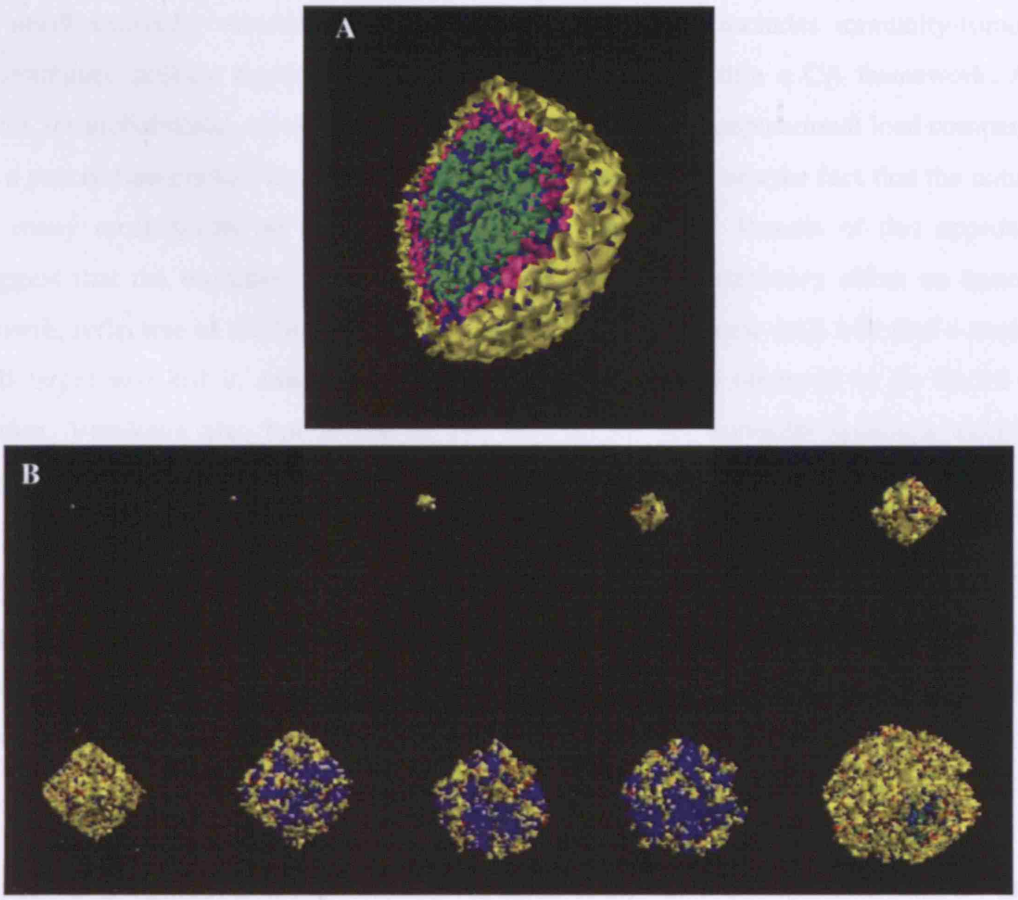


Figure 2.4. **A**. Simulated tumour – colours represent cell cycle phases (yellow, proliferating; red, mitosis; blue, necrotic products; purple, G_0 phase; green, necrosis or apoptotic). **B**. Progression through tumour growth. As can be seen from both views, the morphology is at least qualitatively very similar to the MTS. Stamatakos *et al.* then go on to simulating the effects of radiation on the tumour, discussed more in §2.2.3. The morphology is found to change dramatically as the proliferating cells die and quiescent cells survive. The model does not take into account cell mutation or angiogenesis. Graphics are rendered in specialised software. Pictures taken from Stamatakos *et al.* (1998).

It is important to realise the immediate difference between this approach and the differential equation approach as described earlier – in the CA-like formalism the *local* interactions form the basis of the state of the whole system, which is in fact reflective of

the real system. Even with gross simplifications in the form of local interaction rules the simulation achieves satisfactory agreement to *in vitro* small cell lung cancer MTS with relatively richer spatiotemporal dynamics compared with the O/PDEs presented earlier.

Voitikova (1998) describes a stochastic non-Markovian simulation of immune response to small avascular tumour tissue having a rule-base that includes immunity-tumour interactions, cellular random walks, growth and necrosis within a CA framework. All rules are probabilistic, an approach that is known to reduce computational load compared to a purely fine-grained deterministic method and also underlines the fact that the nature of many mechanisms on the smaller scale are not known. Results of this approach suggest that the immune-tumour interaction may have an oscillatory effect on tumour growth, reflective of the fluctuations in probabilities that immune cells will find a cancer cell target and kill it. Interestingly the tumour geometry is observed to be fractal in nature. Voitikova also boldly claims that this simple yet powerful approach exhibits tumour behaviour simulation better than any O/PDE based model.

More recently, Dormann & Deutsch (2002) use a two-dimensional hybrid lattice-gas CA approach that incorporates cell migration as well as the usual nutrient flow, proliferation and cell necrosis. The lattice-gas 'flavours' of CA are frequently used to model fluid dynamics and specifically contain rules that idealise conservation of momentum and flow of particles. In this case the particles are biological cells whose positions are in flux due to pressure and/or chemotactic gradients. Here the chemotaxis is mediated by a hypothetical signal emanating from near-necrotic cells, which attracts other cells towards them – a rule that is meant to account for the experimental observation of normoxic cells moving towards the necrotic core. Dormann & Deutsch aptly highlight the fact that the purpose of such spatiotemporal modelling is that it offers insights into the mechanisms responsible for collective organisational behaviour at the microscopic level, rather than having a purely phenomenological model that offers relatively fewer insights. The model then quite elegantly shows that the familiar ring structure (normoxic to hypoxic to necrotic) can be recreated relatively easily and therefore proves the point that local interaction-based modelling provides a behaviourally rich and realistic simulation.

Patel *et al.* (2001), using a similar CA hybrid approach, focus on a different target in the microenvironment for tumour growth – that of acidity levels and its effects on invasion. The hybrid model incorporates the local interaction-centric advantages of a CA whilst at the same time utilising the power of differential equations solved on the discrete space/time grid to describe the diffusion of nutrients and H^+ ions in the interstitial space. The difference to other work is the fact that it is one of the very few that actually incorporate the presence of vasculature from which nutrients can reach tumour tissue, albeit randomly distributed in the lattice. Therefore, although compared to and described as early tumour MTS-like growth, strictly speaking this model actually describes vascular growth. The results show the power of the approach – the model showed that an increase in H^+ ions promotes tumour growth and invasion; leading to the prediction that the introduction of vascular density above a certain threshold therefore may actually *decrease* the growth rate since the blood flow reduces the concentration of acidity in the interstitial space.

The T-7 group of Los Alamos National Laboratory[5] (LANL), in particular Jiang, Pjesivac, Freyer and other co-workers, have also taken up these newer mathematical techniques to model MTS dynamics. Like Patel *et al.* (2001) they have used a hybridised approach combining the powers of both CA and PDEs. Their model incorporates the effects of mitosis, mutation, necrosis, nutrient uptake and metabolic waste whilst being able to follow the fate of individual cells, which is not possible with differential equations. Chemical reaction-diffusion dynamics is governed primarily by three PDEs. With this approach the group preliminarily report a strong correlation with experimental data with respect to growth dynamics.

2.2.1.5 Other Growth Related Models

Other models for growth mostly consist of terms for metastasis, which is inherently tied in with angiogenesis (next section) and cellular adhesion. Araujo & McElwain (2004) review recent efforts in great detail, which will not be repeated here. A clear point of

[5] http://math.lanl.gov/

interest is that there is a surge in use of computational techniques such as CA as well as ABM, e.g. the works of dos Reis *et al.* (2001) who employ an agent-based approach to simulate the effects of cellular adhesion on tumour morphology. In summary, local interaction-based models conclusively exhibit more faithful behaviour when compared to the real system compared to traditional techniques.

2.2.2 Angiogenesis Models

Vascularisation of tumours is the most important event preceding malignancy, providing the pipeline for metastases, and is arguably one of the most important factors that determine the overall behaviour of a clinically aggressive tumour (Mantzaris *et al.*, 2004). The fact that tumours can induce nearby capillaries to undergo neovascularisation has provided the impetus for research into molecular targets for therapy (Harris, 1997). The modelling of vascularisation is one of the most complex problems in tumour modelling because of the multiple scales at which all critical events occur. Ranging from molecular interactions to gradients and diffusion through media of differing viscosity, branching and meeting of capillaries and overall tumour dynamics, as well as the complexities of blood flow itself, angiogenesis models must incorporate an exceptional level of spatiotemporal dynamics. Angiogenesis is distinct from the context of the tumour and can be classed as a complex system in its own right (Levine *et al.*, 2001).

2.2.2.1 Basic Physiology of Angiogenesis

In angiogenesis blood vessels are formed from the scaffold vessel structure resultant from a previous vasculogenesis process (Figure 2.5). The basic steps of tumour-induced angiogenesis are as follows (Mantzaris *et al.*, 2004):

1. Hypoxic cells, under the strain of competition for nutrients, produce and emit growth factors generally known as Tumour Angiogenic Factors (TAFs), e.g. VEGF. TAFs can either be angiogenic (e.g. VEGF) or anti-angiogenic (e.g. α-interferon, endostatin) and sometimes both depending on the local state.

2. The endothelial cells (EC) of existing capillaries, which have specific receptors for TAFs, receive the signal that diffuses through the extracellular matrix (ECM). This sets off a chain of intracellular events that eventually leads to cell proliferation (Figure 2.6). The EC then exhibits an activated mesenchymal phenotype (rather than its normal quiescent phenotype).
3. The ECs begin to excrete matrix metalloproteases (MMP), which break down the basement membrane, made of pericytes, and the fibres in the surrounding ECM.
4. As the ECM is broken down it is the leading EC, chemotactically migrating up the signal gradient, that guides a following posterior mass of proliferating ECs. The migration is also influenced by gradients of cellular adhesiveness mediated mostly by fibronectins and integrins. It should be noted that proliferation of ECs only begins to take place 36-48 hours after the initial EC response – before this point ECs are actually recruited from the parent vessel (Pawaletz & Knierim, 1989).
5. Capillary tubes form once ECs begin to re-associate and align. Intracellular vacuoles extend to the poles, forming a hollow tube.
6. The immature vessels are said to become mature when pericytes envelope the newly formed structure. Maturation is a critical step in vascularisation (Arakelyan *et al.*, 2002).

Once the above stages of vascularisation have occurred anastomosis[6], the process by which independent branches join, takes place before complete maturation (Mantzaris *et al.*, 2004). Whilst it is easy to think of this process as diffusion-driven and that anastomosis occurs as a result of following simple diffusion gradients, angiogenesis is still an extremely difficult subsystem to model – the exact processes and mechanisms are still not fully understood. This is illustrated, for example, by the fact that whilst independent branches can loop they can also unexpectedly split and the reasons for this are not known at present (Anderson & Chaplain, 1998).

[6] Synonymous with "anastosmosis" in the literature.

Angiogenesis-related molecular species form the basis of most angiogenesis modelling implementations. In tumour-induced angiogenesis the process becomes somewhat anomalous because of the intensity and volatility of extracellular signals resulting from extensive mutations, aneuploidy and hypoxic stress of tumour cells (Dasu *et al.*, 2003). Morphologically this translates to fenestrae, transcellular holes and convoluted tubules (Papetti & Herman, 2002), illustrated in Figure 2.5b and 2.6c.

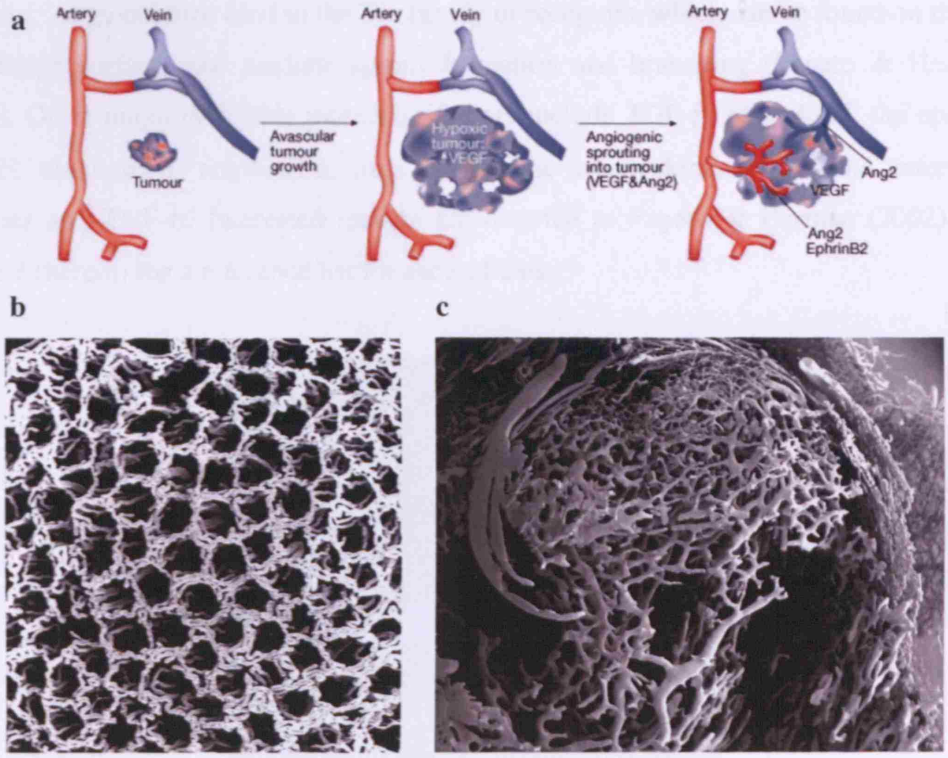


Figure 2.5 a. An avascular tumour is nourished through peripheral capillary networks. TAFs encourage new capillary growth. Image taken from Yancopoulos *et al.* (2000). **b**. Corrosion cast of normal sub-mucosal vessels of the colon. Note the regularity in shape, illustrative of the tight regulation of cellular signals that instigate and coordinate vascularisation. **c**. Corrosion cast of SW1222 tumour xenograft. This irregularity in shape is characteristic of tumours due to the combination of progressive mutation and heterogonous signals from the tumour, which affects EC movement. Images B and C have been taken from the work of Amos A. Folarin (within the same department as the author) who is currently working on modelling blood flow specifically in the context of tumours.

One of the most well-known TAFs is the VEGF isoform family of extracellular proteins. There exists a corresponding family of receptors (most commonly mentioned is the VEGFR-1 receptor) that are expressed on the membranes of endothelial cells. Once

VEGF has bound to its receptor the cascade represented in Figure 2.6 is triggered, resulting in the preliminary stages of angiogenesis. This not only prepares the cells for angiogenesis but also increases the permeability of the vessel, which is in agreement with experimental data suggesting that tumour-associated vessels are extremely leaky (Papetti & Herman, 2002, Kohn *et al.*, 1992). Angiopoietins are another class of extracellular signals involved in angiogenesis. It is worth mentioning that there is a severe under-representation in the literature with respect to models that incorporate the action of these proteins. Angiopoietins bind to the Tie family of receptors, which can be found on the EC membrane surface, and mediate sprout formation and branching (Papetti & Herman, 2002). Other major diffusible molecular factors include TGF-β, FGF, EGF, the ephrins, PDGF, angiogenin, angiostatin, angiotropin, the interleukins, TIMP and interferon families and TNF-α. Interested readers are directed to Papetti & Herman (2002) (and Table 1 therein) for a reference list for each of these.

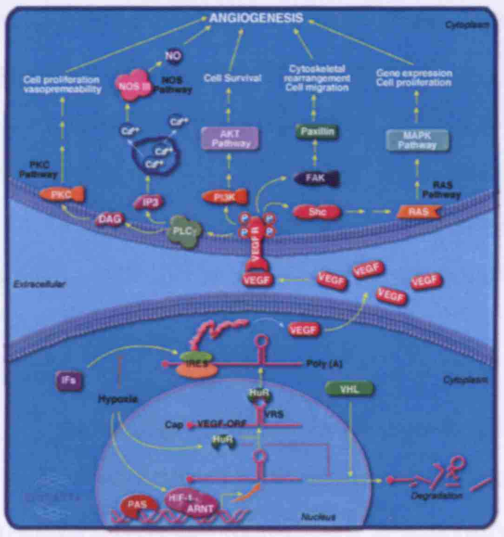


Figure 2.6 Hypoxia-induced TAF production (in this case, VEGF) from the tumour cell (bottom) diffuses to nearby ECs, which receive the signal through a TAF receptor. The resultant cascade results in transcription and translation of genes that will be involved in mitosis and enzymatic breakdown of the ECM (Mantzaris *et al.*, 2004). Picture taken from Biocarta[7].

The ECM is composed mainly of interstitial tissue, collagen, fibronectin, vitronectin, fibrin, von Willebrand factor and other structurally important molecules and plays a

[7] http://www.biocarta.com/pathfiles/h_vegfPathway.asp

critical role in the leading EC's migration (Anderson & Chaplain, 1998). Fibronectin remains static in the ECM, although gradients of fibronectin exist, and plays an important role in cellular adhesion by binding to the integrin family of cell surface receptors. Cadherins are particularly important cell surface molecules, mediating cell-cell interactions. It is therefore clear that though an important role is played by chemotaxis (i.e. VEGF, etc.) there is also the haptotactic response to take under consideration, which many models ignore. The frictional forces that the ECM imposes, due to dense content (i.e. collagen, fibronectin etc.), have a massive impact on the path the ECs take when moving up attractive concentration gradients (Mantzaris *et al.*, 2004).

Finally, the mechanical forces imposed by blood flow itself, which can have a significant influence on vessel formation and maturation, are another important factor in neovascularisation (Papetti & Herman, 2002). Stress in the vessels can induce transcription of PDGF and TGF-β, which affect matrix consistency and vessel formation respectively (Resnick & Gimbrone, 1995). Tumour blood flow models are relatively scarce in the literature and most of the research is conducted in light of existing models of hydrodynamics, a well-developed domain in the field of engineering. When applied to blood flow this is termed haemodynamics. However, the majority of the literature in this case is still applied to cardiac physiology-related haemodynamics rather than tumour-related haemodynamics. This type of modelling utilises complex PDEs and Lattice-Boltzmann flavours of CA.

The brief summary of key molecular, cellular and tissue-level events in angiogenesis presented here highlights the complexity of the process. Furthermore, although many of the molecular species involved have been identified it is generally accepted that still very little is known about the biochemical interactions that are taking place (both intra and intercellular) to satisfactorily explain the rich behaviour of angiogenic tumour systems (Mantzaris *et al.*, 2004). This restricts the modelling of angiogenesis to being merely qualitative rather than quantitative.

2.2.2.2 Modelling Methodologies

The vast majority of evolved mathematical models that incorporate angiogenesis terms do so by characterising diffusion gradients of TAFs and subsequent responses of nearby capillaries. Spatiotemporal resolution is of course very important in these models and this is exactly what is now emerging from the modelling community, although earlier models have still lagged with the traditional differential equation approaches. The angiogenesis modelling literature can be loosely categorised into the stages on which they focus, though the stages are of course overlapping. Many modelling efforts have gone into focussing on the location and time at which ECs begin to respond to TAF signals. Other efforts have focussed on EC migration, the dynamic structure of vasculature, blood flow and turbulence.

Mantzaris *et al.* (2004) further define three basic categories of models for angiogenesis that can be found in the current literature:

1. **Continuum models**: Cellular populations and molecular species are treated as continuous variables. Cells are usually described in terms of densities. This approach invariably involves the use of an O/PDE approach.
2. **Mechanochemical models**: This type of model takes into account the gradients of chemical species in the ECM (e.g. fibronectin) and the migration of cells through the ECM.
3. **Discrete models**: These have appeared more recently, and like their growth model counterparts, include CA-related approaches. Typically this approach can be described as a discrete realisation of a continuum model, as will be portrayed by the works of Anderson & Chaplain (1998). In ABM approaches individual cell fates can be followed.

To summarise, the generalised equations given by Mantzaris *et al.* (2004), which will not be repeated here, are mostly O/PDE models that describe (i) the density of cells in one to three dimensions and (ii) the concentration gradient from source to capillary of TAF(s) in

one to three physical dimensions with appropriate boundary conditions[8]. Diffusion of particles can be modelled as random walks, multiset rewrites (Giavitto & Godin, 2002) or simple Fick's-like PDEs.

More recently, as with growth modelling, hybrid CA and hybrids thereof have become increasingly popular. This should come as no surprise as the CA approach has proved to be extremely apt at capturing not only rich spatiotemporal behaviour, which is imperative when considering vessel structure, but also heterogeneity. The work of Patel *et al.* (2001) described earlier illustrates the power of CA in vascularised tumour growth. Note here, however, that angiogenesis models do not pertain to mere growth models that incorporate vascular presence, as in Patel and co-workers' model. Rather, angiogenesis models include vascular dynamics – i.e. sprout formation, extension and maturation.

Anderson & Chaplain (1998) demonstrate quite elegantly the strengths and weaknesses of both continuous and discrete modelling methodologies, though this was not the specific goal, by first creating a set of differential equations to describe chemotaxis as well as haptotaxis and EC density dynamics. The model is then solved using an extension of the conventional finite difference method in a discrete 2-dimensional CA setting to gain better spatiotemporal insight. The authors highlight the fact that very few models successfully capture rich enough behaviour to be able to predict complex observable phenomena such as the brush-border effect (the increased level of neo-vessel branching when it approaches the TAF source). Moreover the model also incorporates random walks of leading ECs in order to mimic sprout extension (branching, anastomosis and cell proliferation). Once the continuum model was defined the relative effects of haptotaxis and chemotaxis were simulated in two dimensions. It was shown that concentration gradient geometry and magnitude is important, stressing the fact that assuming symmetrical dispersion of molecular signals in a traditional source-sink methodology does not sufficiently model reality. Additionally the simulations show the crucial role haptotaxis, i.e. the interplay of fibronectin, laminin, and other macromolecules, plays in angiogenesis, which is quite a unique development in the angiogenesis modelling

[8] See equations 1 and 2 from Mantzaris *et al.* (2004).

literature. The model can be readily converted into a 3D setting, as demonstrated by Stephanou *et al.* (2005). The 3D model is shown to be quite different both in terms of actual vasculature dynamics and drug delivery, highlighting the fact that any serious attempt at a tumour-level simulation must be performed in a 3D setting. The group reports that the most striking result obtained is that, when particular vascular structures are formed, only less than 3% of the drug actually reaches the tumour – the rest bypasses it. If correct, this of course bears important implications for therapy.

The discrete model that Anderson & Chaplain then formulate incorporates biased random walk EC behaviour under chemotactic and haptotactic conditions. Since there is a lack of data as to what the anastomotic causal mechanisms actually are, the authors incorporate an age (of sprout) and space constraint rule into the CA to model branching and looping. The individual cells in the CA represent ~10μm, or 1-2 ECs, collated into a 200 × 200 grid. The resulting model is shown to enable both qualitative and quantitative comparisons with *in vivo* experiments. As with the continuum model, the effects of haptotaxis and chemotaxis were simulated and again showed that haptotaxis is likely to play a crucial role in neovascularisation – the simulations suggested that without it no anastomosis could be established and only minimal branching could occur.

The CA approach is found to be far simpler to manipulate, especially in the absence of precise parameterisation data compared to the PDE approach. Of course, individual-based modelling techniques also have the distinct advantage of enabling the modeller to track the behaviour of cells individually, which is of enormous value in the context of angiogenesis.

The random-walk approach was also adopted by Levine *et al.* (2001) who also modelled tumour-induced angiogenesis in terms of haptotaxis and chemotaxis, although no formal comparison with *in vivo* angiogenesis is apparent. Levine *et al.* also incorporate biochemical enzyme kinetics (receptor-ligand binding, etc.), making a rich model that can be interrogated right down to the molecular dynamics scale – given enough computational power this type of cross-scale modelling is possible without losing the rich

fine-grained behaviours of the system, which may well be the case for differential equation DE approaches.

The work described above is a good example of how the tumour system has been dissected mathematically into subsystems, e.g. sprout formation and elongation, and studied in a closed fashion. In other words, by investigating the effects of chemo/haptotaxis on sprout fate for example, the focus is purely on angiogenesis, and other parts of the system are ignored, such as growth dynamics of the tumour itself. One must remember that the process of angiogenesis is repeated again and again and can take days to complete. During this time hypoxic regions could be growing or shrinking and therefore TAF signalling would be in flux. However, since the overall growth of the tumour is assumed to be static the modelling of angiogenesis itself has been restricted to prediction of location of sprout formation and subsequent vessel formation with steady TAF diffusion gradients. Moreover, important additional factors, such as the maturation stage of the new vessels would be omitted in closed sub-system modelling. In immature vessels blood flow is rather poor (Papetti & Herman, 2002). If perfusion is extremely poor or the levels of VEGF and other TAFs falls in the vicinity of immature vessels (IV) then the vessel can regress by, for example, mediation of Ang-2 (Holash *et al.*, 1999).

Arakelyan *et al.* (2002) recognise the fact that IV and mature vessel (MV) structure stabilisations are critical in modelling angiogenesis since vasculature has been observed to be a dynamic rather than static structure. This is in sharp contrast to previous models where, once a leading EC has set a path for a new vessel, the vessel remains static and is assumed to be instantaneously mature and perfused. This assumption is made in both continuum models (in the form of densities of vessels) and discrete models. Arakelyan and co-workers therefore set about producing a discrete model to describe not only the angiogenesis processes of vessel formation and maturation but also the simultaneous fluctuation of tumour growth and TAF signalling (no account is taken for the influence of ECM macromolecules). However it must be noted that even though the model incorporates vessel destabilisation it makes the simplification that IVs and MVs actually have the same tissue perfusion efficiency, which is known not to be true, and therefore

enumerate both types of vasculature generically as effective vessel density. The algorithm employed by Arakelyan *et al.* includes a fusion of abstract parameterisations and an extensive Boolean network. A Boolean network is simply a graphical model incorporating "yes/no" decision-making arcs and process nodes and is therefore essentially a Heaviside-step threshold-based system. The algorithm is described as multi-scale since it covers three organisational levels – molecular, cellular and organ-level. The model was tested by performing *in silico* simulation of functional knockout experiments, which would be very difficult to perform experimentally. For example, it was shown that in the absence of vessel destabilisation and maturation tumour and vessel volume follow exponential growth laws. Subsequent incorporation of these behaviours results in fluctuations in IV and MV volumes, and was shown to be in marked agreement with experimental data (although the agreement is qualitative rather than quantitative).

In summary angiogenesis models illustrate the power of individual-based models over traditional techniques. The geometry of vessel growth and its importance to growth is very well suited to individual-based models. Furthermore angiogenesis illustrates the complexity of the tumour modelling problem, consisting of a large number of stimulants and inhibitors with complex interactions over large time and space scales.

2.2.3 Treatment Response Models

Response behaviour is the ultimate goal of tumour modelling since it is in the successful simulation and prediction of critical parameters of this behaviour that will finally lead to clinical improvement. Response models, as a matter of necessity, must include growth aspects of tumours and, preferably, vascularisation aspects. Angiogenesis-response model hybrids have become more prominent in the literature only recently with the arrival of anti-angiogenic drugs. Pathways modelling with respect to treatment response is still relatively scarce – however one can expect that the inclusion of pathway dynamics in response models will become more popular with the continuing explosion of knowledge of cellular signalling and the concept of targeted therapy.

Two kinds of tumour response models can be observed in the literature:

1. **Mechanistic-based response models**:
    a. Chemotherapy (and subsequent immune-) response to tumours and use of therapies to boost immune response. This includes radio-immunotherapy.
    b. Tumour response to radiation therapy.
    c. Dynamic pathway response to the two therapies listed above (discussed in more detail in §2.2.4).
2. **Therapy optimisation** (e.g. dosimetry models): These models focus on exactly how and when therapy should be administered rather than focusing specifically on tumour dynamics.

The same formalisms that were discussed for tumour growth apply here –the tumour volume, density, number of cells and/or tumour radius are the primary outputs of response models although some models also go as far as including metastases. Also, the same temporal pattern of use of formalisms appears in the literature, i.e. first use of DE approaches for avascular tumours and later use of more descriptive computational approaches, such as CA and ABM.

2.2.3.1 Generalised Formalisms

A basic differential equation approach can be expressed as follows[9]:

$$\frac{dn}{dt} = pn_t - k_t n_t \qquad \text{Eq. 2.9}$$

where:

n_t is the number of cells at time t.

p is the proliferation rate of the tumour.

k_t is the therapy effectiveness at time t and so $k_t n_t$ denotes kill rate.

[9] http://www.ap.univie.ac.at/users/ves/cp0102/dx/node126.html

The variable p can be generally determined for tumour cell types and k_t is drug-dependant. This model is phenomenological and includes no spatial dynamics. However, it forms the basis of most response models that use the DE approach.

Equation 2.10 can be used to estimate another term that is extensively used in modelling – the so-called Tumour Control Probability[10] (TCP), which is used mostly in radiotherapy models:

$$TCP = e^{-\sum_{i=1}^{Q} N_i S_i(D)} \qquad \text{Eq. 2.10}$$

where:

TCP is the probability that no tumour cells survive (tumour cure probability).

i and Q denote the total number of cells in the considered 3D space i.

N_i is the initial number of cells.

$S(D_i)$ is the surviving fraction of cells after dose D in region i.

The LQ model is another general formalism for radiotherapy and describes the survival probability of a cell. It reads as follows:

$$S(d) = e^{\left[-\left(\alpha d + \beta d^2\right)\right]} \qquad \text{Eq. 2.11}$$

where:

$S(d)$ is the survival probability of the cell given dose d.

α and β are parameters specific to the cell/tumour and reflect intrinsic radiobiological properties pertaining to sub/lethal DNA damage due to radioactivity.

d is the dose (usually in Gy, where 1 Gy = 1 Jkg^{-1}).

[10] For an in-depth discussion on TCP, its applications and extensions the reader is directed to Zaider & Minerbo (2000).

The above equation can additionally be extended to account for the sensitivity of hypoxic tissue by taking into account the OER:

$$S(d) = e^{\left[-\frac{\alpha d}{OER} + \frac{\beta d^2}{OER^2}\right]} \qquad \text{Eq. 2.12}$$

where *OER* is a constant oxygen enhancement ratio and specific to a particular tumour type.

2.2.3.2 Modelling Methodologies

Immune response and therapy models are abundant in the literature (see Adam & Bellomo, 1996). The works of de Pillis & Radunskaya (2001, 2003) demonstrate the state-of-the-art in this kind of modelling using a combination of traditional approaches with optimal control theory to develop an avascular model (homogenous, MTS-like) that can qualitatively predict *in vivo* tumour dynamics. The model recognises three types of interaction that have previously been modelled separately: (i) interaction between tumour cells and immune system cells, (ii) interaction between normal cells[11] and tumour cells and, (iii) interaction between tumour cells and chemotherapeutic agents. The model therefore effectively idealises competition between immune system cells and tumour cells (and their response to therapy) and the corresponding immune response to the presence of a tumour. Differential equations are formulated that determine the number of normal, tumour and immune cells and the cell kill relation for a therapeutic agent. The resulting model is reported to capture two interesting behavioural characteristics that are observed *in vivo*. Firstly, so-called "Jeff's Phenomenon" is observed – an asynchronous oscillation of tumour growth with respect to chemotherapy dose time. Secondly, and perhaps clinically more interesting, is the model's ability to predict tumour dormancy – a phenomenon that occurs when the tumour shrinks to an undetectable size only to re-emerge to grow exponentially again. Both of these behaviours can be explained by the interactions of the tumour and immune cells. This example alone goes to show the rich and qualitative (and eventually, in the future, quantitative) dynamics that mathematical

[11] Note that the interaction between normal cells and chemotherapeutic agents is not included.

modelling can actually generate which serves to better current understanding. The goal is of course to reduce tumour cell counts whilst keeping normal cell counts within safe bounds by finding appropriate times to administer certain amounts of dose in a given time interval. Optimisation and control theory is applied to achieve this and is extensively tested in more recent papers (de Pillis & Radunskaya, 2003, de Pillis *et al.*, 2005). A drug protocol is thence determined where the tumour cells are eradicated whilst keeping the total normal cell count around the tumour site above 75%, though the oscillations in tumour size observed in the simulation would admittedly not be clinically desirable.

Arciero *et al.* (2004) demonstrates another interesting theoretical approach. This example is particularly appealing because it models an immunotherapeutic strategy that has not yet been tested extensively *in vivo* – so called "small interfering RNA" (siRNA) immunotherapy – and so this scenario depicts a unique opportunity for mathematical modelling to prove itself as a viable technology with which one can satisfactorily predict tumour dynamics. The siRNA molecules are only around 22 nucleotides long and interfere with certain transcripts such as that of TGF-β, thereby blocking gene function. TGF-β is involved in masking the tumour from immunosurveillance, it is also strongly angiogenic and relatively well understood, making it a good target. The proposed model uses a system of only five ODEs that describe immune cell numbers, tumour cell numbers (incorporating the phenomenological logistic function), IL-2 dynamics and the effects of TGF-β on tumour growth and immunosurveillance and subsequent effects of siRNA that blocks TGF-β and degradation of the siRNA. Administration of treatment is found to have oscillatory effects on tumour growth dynamics, just as de Pillis & Radunskaya described earlier.

This touches upon another active area of research in cancer systems biology – that of the effects of drugs on signalling pathways. Surprisingly the number of models that actually incorporate pathway dynamics directly with drug action is quite limited (i.e. both aspects *together* – the literature in terms of the individual aspects are actually substantial). These types of models will be discussed in more detail in §2.2.4, however to make a case in point the work of Charusanti *et al.* (2004) is briefly explained. The model proposed in

this case demonstrates the power of simulations and makes use of traditional ODE approaches to predict the effects of STI-571 (Gleevec) on the Crkl pathway (Figure 2.7) in Chronic Myeloid Leukaemia (CML).

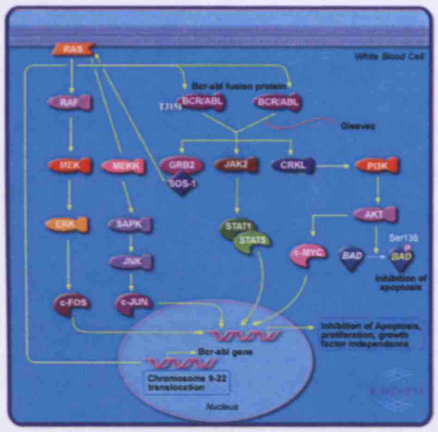


Figure 2.7. Gleevec action on the Bcr-abl oncogene. CML is probably the only type of cancer that is associated almost exclusively with a single type of mutation, i.e. 9-22 chromosome translocation (the "Philadelphia Chromosome"). Deregulated phosphorylation, due to this mutation and caused by the over-active Bcr-Abl gene, causes certain signalling pathways to be constitutively switched on (e.g. proliferation pathways, not shown) and others to be switched off (e.g. apoptosis, bottom right of figure). Picture taken from Biocarta[12].

The ODEs consider the concentration differential of certain populations of molecular species over time and the Gleevec pharmacokinetics (compartmental model of intraperitoneal cavity, blood and tumour). Even though no specific attention is paid to the spatial arrangement of tumour cells and normal cells, the model is considered satisfactory since cancers originating from the haematopoietic pool in the marrow, like CML, are only quasi-solid. ODE approaches in this particular situation therefore are appropriate since no spatial dynamics need to be modelled (unless drug delivery itself is the focus)[13]. The resulting simulation suggests an important previously unrecognised dynamic of Gleevec treatment, which was later experimentally shown to be a real behaviour – during blast crisis the clearance of Gleevec from cells is very rapid and therefore the effectiveness of the drug is reduced. Here is an excellent example of *in silico* prediction

[12] http://www.biocarta.com/pathfiles/h_gleevecpathway.asp

[13] Spatiotemporal modelling of these types of tumour will be very difficult compared to solid tumour modelling and hence the next step in modelling in the years to come will involve the consideration of quasi-solid tumour dynamics.

and experimentation – the simulation immediately suggests that mechanisms of clearance must be addressed to enable Gleevec to function with greater efficiency.

Modelling has also established its place in therapy response with respect to angiogenesis[14], and it has already been explained why targeting of angiogenic processes has become an important topic in oncology. Though the focus in Arakelyan (2002) is not entirely on the effects of drugs on tumour dynamics, by incorporating ODEs for drug action (VEGF and Ang-1 production inhibitors) into the model potentially important qualitative behaviour can be observed – their simulations suggest that, for an aggressive tumour that has relatively low sensitivity to anti-angiogenic drugs, monotherapy is not sufficient to eradicate the tumour. The model showed that only combinatorial therapy successfully killed all tumour cells. Stoll *et al.* (2003) take a slightly different approach by considering the contribution of endothelial progenitor cells (EPC) from the bone marrow to angiogenesis as well as the contribution of ECs in the vicinity of the tumour. Stoll and co-workers establish a set of ODEs that describe vascular densities and flow of EPCs into the region and re-parameterise the equations to account for drug action. Drug actions considered include the effects of therapy that target EPC migration, growth factor and angiogenic factor signalling, chemotherapy by anti-angiogenic scheduling and combined therapies. The model is found to be in agreement with clinical data.

Whilst the works briefly described above take a semi-mechanistic approach to modelling, there are still those models that are almost totally phenomenological such as those presented by Ubezio (2004). In this model cell cycle features are incorporated into a simple compartmental approach and curve fitting to cytometry data is used to parameterise the ODEs. Drug effects are then simply modelled by stochastically altering the behaviour of the compartments, e.g. by giving a cell a probability of being blocked in certain stages of the cell cycle.

---

[14] A whole host of response-to-angiogenesis targeted therapy models exist but cannot be summarised here due to space constraint. The reader is referred to the works of Araujo & McElwain (2004) (Araujo & McElwain, 2004) for a good review and, for specific models, D'Onofrio & Gandolfi (2004), Plank & Sleeman (2003), Scalerandi & Sansone (2002) and McDougall *et al.* (2002).

In the immune response modelling area CA approaches are less prominent than traditional approaches – this could be because response models often focus on the drug delivery aspect, dosimetry, direct effects on growth and related optimisations as opposed to individual-based understanding, e.g. cell to cell interactions. Here again however it can be argued that delivery and diffusion of drugs through the (dynamic) vasculature and heterogeneous flow through the ECM all have observable impacts on tumour dynamics and cannot be modelled by traditional methods. Therefore one can expect that CA or ABM models to take up a more prominent role in this area.

McDougall *et al.* (2002) describe an extension of the model by Anderson & Chaplain (1998) by incorporating a perfusion function into the angiogenesis model thereby formulating a 2D simulation for drug delivery in a vascularised tumour. Interestingly the flow model was actually adapted from CA-like petroleum flow models, which simulated the "perfusion" of petroleum through pore networks in solid rock. Though this formalisation only models perfusion around the tumour rather than through it, some very interesting results were procured pertaining to delivery strategy. Having simulated two chemo-agent introduction schema (one-off bolus injection and continuous infusion) it was predicted that large one-off doses may sometimes lead to dilution in the over-perfused regions and therefore actually never reach the tumour, which obviously would have serious implications for therapy. One must remember, however, that this model does not take into account maturation of vessels, i.e. vessels are assumed to be immediately mature. This is of course not concordant with the real system.

Many other chemotherapy response models exist that focus on different clinical aspects. Pinho *et al.* (2002), for example, formulate a continuous model where the interactions between normal and cancer cells are considered in the context of primary and secondary tumour sites, i.e. metastases, and interactions with a chemotherapeutic agent. There are many different clinical strategies for optimal therapy and therefore the optimisation problem has become very complex consisting of many sub-problems. For example, there are those models that focus specifically on pulsed therapy as well as drug resistance, which have become increasingly important issues. Lakmeche & Arino (2001) develop

such a model, focussing on pulsed therapy and chemotherapy-induced drug resistance by utilising complex sets of both ODEs and discrete algebraic equations. Similarly Jackson & Byrne (2000) describe a PDE model that accounts for vasculature as well as effects of drug resistance.

Optimisation of treatment has been discussed with respect to chemotherapy above. However the vast majority of literature that deals with this branch of modelling is devoted to radiotherapy. Often the modelling of radiotherapy seems to employ a different line of enquiry with respect to other types of models – rather than focusing on tumour dynamics the vast majority of models are phenomenological, which derive quick and simple solutions to treatment problems such as fractionation, hence the TCP and LQ models. Only relatively recently have the concepts of tumour dynamics and response been fused. Traditionally the optimisation approach always takes into account four main tenets, the so-called "Four R's" described as follows (Stewart & Traub, 2000):

1. **Repair effects**: a lethal dose of radiation causes a double-strand break in the DNA of the recipient cell, causing the cell to die. Single strand breaks (sub-lethal events) can also occur but these are repairable by the recipient.
2. **Reoxygenation**: hypoxic cells become reoxygenated either by cell migration or vascularisation.
3. **Redistribution of cell cycle**: quiescent (hypoxic) cells are relatively harder to kill. Once these cells re-enter the cell cycle they are more susceptible to therapy.
4. **Repopulation**: due to the first three R's.

The work of Stamatakos *et al.* (1998, 2001) has already been discussed in the context of growth in §2.2.1. However that research group is also actively modelling the effects of radiation therapy on avascular tumour growth. Recently they described the utilisation of a Monte Carlo approach to modelling in 3D which is very similar to that described earlier (except that the cell cycle model is more detailed). The model is CA-like in nature and simulates avascular growth; excellent agreement with experimental data is reported (Zacharaki et al., 2004). Since space is discretised the LQ model is applied to individual

voxels, which themselves represent cells – in fact the same methodology is applied by Stewart & Traub (2000), explained above. The radiobiological effect is modelled as a function of the cell cycle stage in which those cells are situated. Although this paper does not allude to any formal optimisation approach to find the best course of treatment, apart from the declaration of four fractionation regimes, the simulation predicts that all fractionation tests fail to eradicate all tumour cells. The overall simulated tumour dynamics exhibit good agreement with experimental results. A simple iterative process to this simulation can yield a good *in silico* optimisation strategy.

A similar approach is also adopted by Borkenstein *et al.* (2004). The model takes into account cell cycling and response to radiation, as explained already for the models above, but also is one of the few models that incorporates the effects of angiogenesis and subsequent growth. This makes it one of the most powerful radiobiological response simulations available currently. Since the response formulae are applied in a discrete fashion a heterogeneous population of cells with differing sensitivities and responses to radiation according to cell state and immediate environment can be considered. Simulations revealed interesting characteristics of growth after fractional treatment, highlighting the importance of timing and repopulation of quiescent cells. Interestingly the results were quite similar to that produced by the Stamatakos group – an accelerated fractionation dose was predicted to give best chances for tumour control. Another intriguing result that the Stamatakos model could not have simulated, however, was the effect of angiogenesis to response – since vascularisation decreased hypoxia the tumour became more sensitive to treatment but only if proliferation of normoxic cells was fast. If proliferation was slow then the therapeutic benefit of having reduced hypoxia was eliminated and angiogenesis led to decreased response.

2.2.4 Dynamic Pathways Models

The field of intra/intercellular molecular pathway research and associated mathematical and computational methods embodies the core of the systems biology field. This is because ultimately the behaviour of pathways that will define the behaviour of the cell,

which in turn contributes to the behaviour of the entire system (Cho & Wolkenhauer, 2003, Zhu *et al.*, 2003). Pathways are currently an extremely active area of research and constitute an enormous body of literature.

Generically, three types of intracellular pathways can be defined from the literature that is qualitatively different in nature:

1. *Metabolic networks* which have been dealt with for many years and involve enzyme/substrate biochemical reactions.
2. *Gene networks* which are networks that are controlled both on the proteomic level (e.g. protein-protein repressors) and on the DNA/RNA levels (e.g. DNA-protein repressors and self-looping RNA), and are sometimes synonymous with the third kind of pathway in the literature.
3. *Signal transduction pathways*, which are dynamic in structure (especially in systems where mutation is a key factor) and in molecular flux and are involved in signalling of extra/intracellular stimuli, ultimately resulting in gene transcription (i.e. gene networks).

Intercellular signalling is generally considered comparatively less complex and has already been discussed to some degree in terms of diffusion in the angiogenesis modelling section and so will not be discussed any further here. The methodologies discussed below will focus on signal transduction as this has been predominant in the cancer-related modelling literature.

2.2.4.1 Modelling Methodologies

With the arrival of high-throughput technologies, such as microarrays, two types of distinct algorithms can be observed in the literature (Kitano, 2002) – (i) data mining of high-throughput datasets to predict and define causal relationships between genes and proteins to build up conceptualisations of pathways; (ii) simulations of established conceptualisations. A vast amount of literature is devoted to the former, but will not be

discussed here in any detail in light of the main goals of this project[15]. Stochastic models that are derivatives or relatives of Bayes' theorem dominate this type of modelling (Baldi & Long, 2001, Eddy, 2002, Troyanskaya *et al.*, 2003, Friedman *et al.*, 2000).

Computational methods have been extensively used in pathways research, both in terms of visualisation and simulation, which can be seen by the appearance of computational methodologies and techniques such as Systems Biology Mark-up Language[16] (SBML). The number of databases[17] that currently exist, e.g. Biocarta[18], KEGG[19], DIP[20] and many others, reflects the amount of research that has gone into representations and visualisations[21]. Dynamic pathway models can be very difficult to compile and verify, which is possibly due to the lack of data to which models can be fitted (Cho & Wolkenhauer, 2003). As will become apparent in the next few paragraphs the vast majority of dynamic pathway models are based on ODEs, which of course need to be parameterised. Whilst microarray data, though extremely noisy, provide at least some data there exists no such comparative technology as yet for protein measurements, though some attempts have been made at high-throughput proteomics (Naistat, 2004).

Cho & Wolkenhauer (2003) highlight the importance of spatiotemporal modelling of the individual cell. Not only are molecular interactions highly non-linear, they are highly organised both in space and time, which means that serious models must also take into account the spatial organisation of organelles within the cell as well as the molecules themselves. Zhu *et al.* (2003) have responded to this problem with a call for new methods that can tackle the transient molecular interaction capacity and flexibility of gene regulatory networks and signalling pathways in general. ODE approaches are said to be an inappropriate strategy since the network itself is evolving *in situ*. They make a special

---

[15] Usually these algorithms themselves are not mathematical models of the system and therefore are not of interest to the main objectives of this project. However, often the outputs of these algorithms, e.g. possible protein networks, do represent models of the system in a more direct way.
[16] http://sbml.org/index.psp
[17] For an excellent review on pathway databases see Schaefer (2004).
[18] http://www.biocarta.com/
[19] http://www.genome.jp/kegg/
[20] http://dip.doe-mbi.ucla.edu/
[21] Visualisation methods are not discussed here – the reader is directed to Kitano (2003) for a review.

distinction for metabolic networks and claim that these are relatively static (assuming no mutation). Crucially, Zhu and co-workers put forward a software engineering project that involves the integration of different types of model, i.e. continuous versus discrete, stochastic versus deterministic, qualitative versus quantitative, to build a hybrid platform[22] on which one can simulate the spatiotemporal evolution of dynamic networks (each integration is programmer-defined). The so-called Grid Cellware project presents a set of optimisation algorithms and solvers and exploits a parallel computing infrastructure to solve user-defined models (usually DE-based).

The Michaelis-Menten (MM) equations are a fundamental formalism for enzyme/substrate kinetics and essentially reduce to a simple ODE (not covered in detail here because the role of these models is not as prominent as others in tumour modelling). Given the reaction between an enzyme (E) and substrate (S) that give the product (P):

$$[E]+[S]\underset{k_{-1}}{\overset{k_1}{\leftrightarrow}}[ES]\xrightarrow{k_2}[E]+[P] \qquad \text{Eq. 2.13}$$

The MM equation, which gives rate of change of product in time, can be expressed as:

$$\frac{d[P]}{dt} = V_{\max}\frac{[S]}{K_m+[S]} \qquad \text{Eq. 2.14}$$

where:

[P] and *[S]* are the concentrations of product and substrate respectively.

V_{max} is the optimum enzymatic kinetics on S.

K_m is the so-called MM constant and represents S at $V_{max}/2$.

An interaction network can then be described as a contiguous set of MM equations.

[22] Cellware, http://www.bii.a-star.edu.sg/research/sbg/cellware/index.asp

Signalling networks are modelled slightly differently but almost all still use the differential equation approach to describe pathway dynamics. Tyson *et al.* (2001), for example, apply different types of (mostly) differentials to certain aspects of cell physiology – genetic regulatory circuits described as ODEs or Boolean networks; spatially-oriented signalling by PDEs and cellular automata; functional or integro-differentials for time delays and spatial averaging; stochastic models when there are small numbers of molecules. Cho & Wolkenhauer illustrate the use of differentials with respect to a simplified version of the NF-κB signal transduction pathway. This particular pathway plays a central role in control of proliferation and apoptosis and is therefore sometimes found to be constitutively "switched on" in many tumour types. The general rate equation used is as follows:

$$\frac{d\mathbf{m}(t)}{dt} = \mathbf{h}\left[\mathbf{m}(t),\mathbf{k}(t),\xi(t)\right] \qquad \text{Eq. 2.15}$$

where:

$\boldsymbol{m}(t) = [m_1(t),m_2(t)\ldots m_p(t)]$ and $m_i(t)$ denotes the concentration of the i^{th} molecule in the network.

$\boldsymbol{k}(t) = [k_1(t),k_2(t)\ldots k_p(t)]$ and $k_j(t)$ denotes the j^{th} rate parameter.

$\xi(t)$ denotes an uncertainty (noise) function at time t.

Most differential based models are derivations of this basic ODE[23]. Accordingly Schoeberl *et al.* (2002) formulated an EGF signal transduction pathway model, which is still one of the most well defined dynamic pathways for tumour biology. Initial values are of course needed before the network can be simulated. This sort of model has huge potential – one can predict the behaviour of the pathway given certain conditions and therefore begin to develop drug-targeting strategies. Schoeberl and co-workers perform a number of simulations and describe molecular dynamics that would lead to deregulated cell proliferation, e.g. a simple increase in the number of EGF receptors. Similarly, Wiley *et al.* (2002) also describe mitogenic activity by modelling the increase of internalisation

[23] Although many implementations do not include the uncertainty function.

of the EGF receptor. Gaudet *et al.* describe a similar methodology for the TNF-α pathway[24]. However, to the best of the authors' knowledge, no formal attempt has been made to extend these particular models to describe the dynamics of the cell or host system (e.g. tumour), which would require model integration. For example a growth model can be developed by linking these pathway dynamics to an individual-based model of growth, i.e. each cell implements the pathway model. However such a linking has important implications for scalability. This angle on model integration is discussed in Chapter 4.

Another notable effort, and extension of the EGF pathway described above, is reported by Miller & Zheng (2004). It has been argued that ionising radiation can inadvertently cause the over-expression of both MAPK and TGF-α signalling pathways and therefore contribute to sustained proliferation. The authors therefore modelled the autocrine signalling induced by radiation-exposed tumour cells. MATLAB[25] is used to simulate the network of 148 chemical reactions and 104 ODEs. The resulting model agreed with experimental data, at least for short-term radiation response. Yet again however, the effects on (dynamics of) the whole tumour system are not considered in any serious way – the pathway model is considered in isolation though it does seem very well suited to be a component of a therapy model, such as that described by Stamatakos' group described in §2.2.3.2. Such an integration would yield a simulation that is far more representative of a clinical case.

Stochastic methodologies are also quite popular in this area. Wolkenhauer *et al.* (2004) reviews the latest methods including the popular chemical master equation, Gillespie algorithm and τ-Leap method – all of which are used extensively in the literature. Meng *et al.* (2004) strongly advocate the use of stochastic methods by arguing the stochastic nature of natural systems such as signal transduction and therefore the inaptness of deterministic models. A distinction in the types of stochasticity is established by virtue of origin – extrinsic (randomness from outside the system) and intrinsic (randomness

[24] http://www.sbi.uni-rostock.de/docs/p_sensitivity_tnf.pdf
[25] http://www.mathworks.com/

generated from within the system). Meng and co-workers then set out formalisations for these types of stochasticity. These stochastic models however are computer-intensive and so Lok (2004) complements the stochastic modelling appraisals by reviewing parallel computing and shortcuts for the chemical master equation-based stochastic algorithms.

It can be argued that the models explained thus far are fundamentally flawed in that the pathways themselves exhibit static topologies. To tackle this problem Shmulevich *et al.* (2002) develop a Boolean/Bayesian hybrid model that fuses the conditional stochastic concepts of Bayesian networks with the simplicity of Boolean networks in a so-called Probabilistic Boolean Network (PBN) formalisation. The mathematics is quite complex and not shown here. However, simply put, a previously fixed topology of genetic regulatory networks are given a choice of reaction functions, yielding a more dynamic network. To the best of the author's knowledge such an implementation for signalling networks is not yet present in the literature though the method does seem to be flexible enough to do so. However, the method as yet remains untested against biological data.

Various other methodologies have also been suggested in recent years, many coming from concepts in computer science. Particularly, ABM has become more popular (Fisher *et al.*, 2001; Chen *et al.*, submitted). This permits spatio-temporal modelling and single molecular tracking akin to CA. Petri Nets are also becoming a popular method (Zevedei-Oancea & Schuster, 2003), as well as knowledge-based reasoning approaches (Baral *et al.*, 2004).

Only very recently has there been any convincing attempt at directly linking pathway dynamics to tumour dynamics. Perhaps the first concerted effort in this respect has been presented by Christopher *et al.* (2004) who employ a "forward-modelling paradigm" to mine through data to first build up a pathway for proliferation (Description Cell Language for representation and ODEs for dynamic description) and then apply it in the context of a cancer cell. The proliferative potential of the cell can then be applied to a wider context of a heterogeneous population of cells (i.e. a tumour) for a tumour growth simulation and/or cell level drug response simulation. The group use the Virtual Cell

platform for simulation, which will be discussed later. Jiang *et al.* (2005) alternatively use a Boolean network to model the behaviour of the cell cycle in response to oxic conditions and hence the effects on tumour growth. With the continuing rise in access to high power computing resources such multiscale models are expected to become more prominent.

2.2.5 Other Models

The review given above is by no means exhaustive with many more techniques only just emerging, e.g. the use of fractal mathematics in vascular networks (Gazit *et al.*, 1997) and drug delivery (Dokoumetzidis *et al.*, 2004). Model focus is also diversifying with increasing attention being given to point mutations (Natarajan *et al.*, 2003) and chromosomal aberrations (Frigyesi *et al.* 2003). Finally, there is a plethora of modelling papers that focus on classification or clustering of some kind of data and then deal with tumour prognosis or diagnosis in some way. None of these models deal with the dynamics of the tumour itself and can be categorised as phenomenological. In fact a huge number of models have been developed in response to the availability of large datasets like those offered by microarray experiments and other molecular-based techniques. For example, Catto *et al.* (2003) use a neuro-fuzzy approach in which linguistic (i.e. qualitative data) could be used to classify patients into prognosis groups. The group report an eye-opening 88-95% efficiency.

2.3 Summary

A whole range of models and techniques that appear in the tumour modelling literature have been discussed. The inadequacies of traditional techniques, i.e. DE models, over computational methods such as ABM and CA have been highlighted. The tumour modelling literature can be dissected by the target system the models attempt to emulate, i.e. growth, response, angiogenesis and pathway models – though it must be pointed out that there is significant overlap across these sets.

2.4 Conclusion

Many different aspects of tumour dynamics, with a diverse collection of methodologies, have been modelled but there is no unifying principle between these models. There is currently little or no scope to reuse these models without extensive redesign and development, especially when the complexity of semantics can be so varied from model to model. To do so would require model integration technologies. The survey made here will help to highlight the points at which such integrations can be made, especially from the viewpoint of model behaviour, which will be further explained in subsequent chapters.

3. Complex Systems Science and Systems Biology

"Systems biology aims at describing biological processes as interactions of molecular components structured by regulatory wirings..." (Alberghina & Colangelo, 2006)

"Systems biology is an academic field that seeks to integrate high-throughput biological studies to understand how biological systems function."(Wikipedia)

"A systemic approach to biology ought to put the emphasis on the entire system; insofar as it is concerned with components at all, it is to explain their roles in meeting the needs of the system as a whole."(Cornish-Bowden, 2006)

"...systems biology addresses the missing links between molecules and physiology."(Bruggerman & Westerhoff, in press)

"...(there seems to be a consensus that) systems biology will progressively complement the conventional mode of study by facilitating the understanding of biological networks and mechanisms in terms of their dynamic system behaviour on different levels of organization." (Dubitzky, 2006)

3.1 Chapter Objectives

- To detail the most salient points regarding complex systems, their behaviour and properties, with a focus on the tumour systems, and define related concepts.
- To introduce the concept of local interactivity in multi-agent systems as a direct explanation of behaviour of complex systems.
- To describe systems biology from a systems-theoretic point of view and review the state of the art in terms of modelling technologies that are used in systems biology.

3.2 Brief

Model integration strategies are the focus of this thesis. To formalise these strategies a detailed understanding of the nature of the very system that is being modelled is required.

The field of systems biology, one of the fastest growing fields, strives to do precisely this – though opinion over its direction is quite mixed, as the quotes above show. In fact systems biology is central to the topic of model integration (in the application of tumour modelling given here) since it provides the basis on which an integration strategy must be placed, i.e. what are the (sub)systems involved; how do they interact; what are their components; what is their behaviour? A systemic view, according to Cornish-Bowden (2006) above, should strive to answer these questions. Is this the direction in which the field is actually going? These questions will be explored concisely in this chapter.

Accordingly, tumours will be examined as complex, adaptive and dynamic systems. Based on this, the nature of modelling methodologies will be formalised in the next chapter, which will make the challenge of model integration clear and will make it possible to formalise solutions.

Definition 3.1: State

A state is a quantitative or qualitative variable of the system that defines some condition of being. A state can also be defined as a specification of a system or process at a particular instant (Wolkenhauer, 2001). For the purposes of explanation in this thesis, the term is also applicable to abstract concepts that do not relate directly to any physical components.

Definition 3.2: Behaviour

Behaviour is a defined pattern of state changes wherein the said state(s) may refer to both environmental and system states (a distinction between system and environment is made in Definition 3.6). A pattern of state changes constitutes a regular configuration, which can be identified (observed) specifically, of state changes over time (or other parameter), possibly in relation to or in conjunction with other states, within some predefined limits. Behaviours will often be referred to as "non-linear" or "complex", which will be discussed further in following sections.

Diverse aspects of biological systems, from the underlying reasons of their complexity to overall behaviour, have been subject to debate and investigation in recent years. This is embodied in the large complex systems research literature and the impact of this research field and system modelling is enormous – some benefits have already been discussed in the previous chapter. However, the formalisation of concepts pertaining to the characteristics of complex systems is still an active field of research.

Relating certain key characteristics of complex systems to how they are typically represented in models *in silico* has direct implications on the engineering of model integration strategies[1]:

- **Complexity in behaviour**: States of the system as well as individual models that mimic the system can be highly nonlinear (non/linearity will be defined more precisely shortly). The integration of models must preserve the behavioural complexity of the original individual entities.
- **Complexity of component parts**: As will be explained in the following sections, complex systems are composed of components that may well exhibit a degree of complexity themselves. It may be necessary to make this layered complexity explicit in a model integration platform.
- **Scale**: Following from the previous points of layered complexity and composition, system components (and their models) on a fine scale can contribute significantly to the behaviour on a larger scale (of both time and space).
- **Compartmentalisation**: Modelling often involves compartmentalisation of the system, which can be based on both real compartments and artificial ones, to simplify the modelling process. A model integration strategy must respect any compartmentalisation that exists within the original models since these constitute implicit assumptions about what the models represent.

[1] A full definition of model integration from a formal perspective is given in Chapter 4, whilst engineering a platform from a software engineering perspective is discussed in Chapter 5.

- **Environmental context**: Complex systems often interact with the environment, which can affect overall behaviour (though the environment itself is seldom modelled as seen in the previous chapter).

The following sections review these fundamental characteristics within the framework of complex systems theory. They will be discussed with respect to their applications to biology and, in particular, tumour modelling and, eventually, model integration (next chapter).

3.3 Dynamic Complex Systems

Most biological systems, including "macro-systems" such as those found in ecology, are said to be dynamic complex systems (Giavitto & Godin, 2002). Although there are a small number of shared characteristics amongst different complex systems (CS) a succinct definition of a complex system is difficult to articulate since there is still no general consensus. An initial definition of a CS can be given as:

1. A system that exhibits behaviour that is often non-linear due to a large number of interacting components. The interactions between the components, e.g. cells, may well be non-linear too.
2. A system whose behaviour cannot be predicted merely by investigating the behaviour of the individual components in isolation.

Definition 3.3: Component *(of a complex system)*
A component is any part of a complex system that can be reasonably compartmentalised, either by physical or abstract partitions, on the basis that it forms a functional (but not necessarily function-critical) part of the whole.

Definition 3.4: Linearity *and* ***Non-Linearity***
Linearity, mathematically, is defined as a behaviour that satisfies the properties:

$$\text{additivity}: f(x) + f(y) = f(x + y)$$

$$\text{homogeneity}: a \cdot f(x) = f(ax)$$

Neither of these properties is present in pure non-linearity: consider the combination of behaviours in complex systems as mathematical functional outputs, as defined above, which produce other observable behaviours where additivity and homogeneity do not hold (e.g. consider the signalling pathways of all the different molecular species within a cell; even when taken collectively this cannot explain the overall behaviour of cell). To define non/linearity more clearly, a linear behaviour, in terms of system states, can be predicted with knowledge of any previous state and then aggregating a fixed (or fixed functional) amount over a parametric index (usually time) to the desired index, whereas this is not always possible with non-linear behaviour. In the case of complex systems non-linearity is a key property (Moiola, 1994; Bar Yam, 2004; Wolfram, 1988). In this thesis, unless otherwise stated, the term "complex behaviour" is used interchangeably with "non-linear behaviour".

Open complex systems invariably interact with the environment(s), which provides a context to their behaviour through feedback mechanisms. In many cases the environment is also considered complex, e.g. the tumour in the wider context of the surrounding tissue and body. Conversely, *closed* complex systems have no such interaction and are therefore said to be independently complex, though the existence of such systems is arguable. Whole organisms are often presumed closed despite the fact that, of course, there is ample interaction with the environment. In reality closed systems do not really exist but are in fact used in modelling scenarios to simplify the complexity problem.

Definition 3.5: Interaction

An interaction in the context of complex systems can be defined as any communicative action between two or more entities where the entities can be either physical objects, such as molecules or cells, or abstract concepts, such as behaviours. The medium of communication is an abstract concept when the entities involved are not directly interacting physically, e.g. a causal interaction within a chain of events. The three main types of interaction described in this chapter are:

1. ***Physical*** *interactions, e.g. enzyme-substrate reaction.*
2. ***Informational*** *interactions where two entities exchange information without physically interacting. This is an artefact of simulating physical systems where the actual physical interactions are not specifically simulated but metamodelled (defined in §4.3.1.3) such that state changes in one entity affect state changes of another purely by mutual observation. Informational exchange will be used to simplify the way one can describe complex physical interactions where minute interactions need not be modelled specifically.*
3. ***Causal*** *interactions are similar to informational interactions in that state changes of entities, which can be linked into networks (where vertices represent entities and edges represent directed causal interactions), affect state changes in other entities. The difference is that a causal interaction is not simultaneously bidirectional, i.e. it is not an exchange via mutual observation, though feedback is possible. See Definition 3.10.*

Definition 3.6: Environment

If an artificial division can be drawn such that it encapsulates the subset of all behaviours and entities of the system of interest; the remaining subset of behaviours and entities in the universe of discourse can be considered environmental.

Definition 3.7: Universe of Discourse

The entire set of considered entities, variables, interactions, etc. in any given modelling situation.

3.3.1 Properties of Complex Systems

A complex system in its entirety represents behavioural dynamics that are the result of interacting components, which are sometimes referred to as agents, on many spatiotemporal levels.

Definition 3.8: Agent

"Agent" has a varied meaning in the computer science literature depending on the context (Weiss, 1999). In this thesis, unless otherwise stated, an agent can be considered an entity/component of a complex system that is assumed to have some degree of autonomy of functionality within the system (as well as across systems via the environment) (Chen et al., submitted). Typically an agent is considered to interact locally within the system, though in an abstracted model this need not be the case.

Besides agents, complex systems are frequently composed of, or interact closely with, other subsystems that are often complex and dynamic in their own right[2] (Bruggerman & Westerhoff, *in press*). A tumour is a complex system demonstrating highly non-linear behaviour (Deisboeck *et al.*, 2001), which can be said to be a function of the interaction between many subsystems under varying environmental conditions, e.g. cells, microvasculature angiogenesis and ECM. Complex systems science must therefore not only encompass the scale between the individual component (agent, subsystem) and the whole (system) but also must capture an understanding of the dynamics of interactions within the system and environmental feedback (Bourgine & Johnson, 2006).

A distinction that must be made before further exploring properties of CS is that of a *complex* system from a mere *complicated* one. Often one thinks of a complex system as something that has many interacting components. Indeed this could also be a qualification for complicated systems but it does not go far enough to define complexity. Similarly, behaviour may well be observed to be non-linear in complicated systems too albeit to a smaller degree. One of the most prominent and observable attributes of a complex system that sets it apart from other systems is that of emergent[3] phenomena (Bar-Yam, 1997). An emergent behaviour or property by definition is a phenomenon that cannot be predicted

---

[2] However this does not need to be the case – complex systems can also be made up of many simple components. Conversely, a simple system can be made up of components that are themselves complex, e.g. the earth contains many complex systems but its behaviour as a planet in the solar system is comparatively simple.

[3] Emergence is an oft-abused pseudonym for complex behaviour that is not easily explainable. Emergence shall be treated strictly as per the definitions given in the main text to avoid any argument about its actual relevance outside of the context given here.

merely by examining the parts of the system by themselves – it is in evidence where the whole really is greater than the sum of its parts (Johnson, 2001) and is therefore parallel to the concept of non-linearity, i.e. non-additivity and non-homogeneity. For example, consciousness itself is considered to be an emergent phenomenon arising from non-linear neuronal activity (Mizraji, 2004). Emergence is a difficult property to gauge quantitatively. It is also very difficult to generalise the most favourable conditions for such a phenomenon to occur, however it is generally agreed that a complex system whose components are many must have the right balance of interaction between components such that interactions are not so sparse as to render the system completely disorganised and not so much that the system becomes too orderly and rigid (Coffey, 1998; Johnson, 2001).

Prior to further discussion one must note that a great deal of debate surrounding emergence in the literature is not only due to differing philosophical viewpoints but also the sheer variety of definitions of so-called emergent phenomena. Emergentists Charlie D. Broad, John Stuart Mill and Samuel Alexander, as well as others over the last century, had set down much of the philosophical foundations of this subject. Based on these works one can tentatively define emergence in a general form (Definition 3.9).

Definition 3.9 Emergence

A property is emergent if it cannot be deduced from the behaviour of the system's components, even with the most complete knowledge of those components, when those components are studied in isolation or in other wholes (Boogerd et al., 2005). In other words, components and their interactions must be considered ***together****, which make up the whole (system). Later explanations and theories of emergence have included the notion of flows of causality in the system, as will be explained shortly.*

Note that the definition for emergence suffices for the work presented here only and is not representative of all views. The present and most popular views are derived largely from that of Alexander from which the concepts of weak and strong emergence have come to light (discussed shortly).

Definition 3.10: Causality

Causality can be defined as a concept by which physical or abstract events occur or properties and/or states are changed as a result of other physical/abstract events, properties and/or states (i.e. causes) (Sowa, 2000). In other words there is an actual dependence of effects, or probabilities of effects, on causes. Formally, A is the cause of B if and only if bringing about A would be an effective way of bringing about B (Nagl et al., 2007). This is termed agency-oriented view of causality. Note that this is one of many views, however the agency-oriented view is most relevant to the work here and suffices so far as (i) model integration and its implications are concerned (Chapters 4 and 5) and (ii) its medical applications are concerned (i.e. medical reasoning). Extending this to the context of complex systems: Any event, property or state that is deemed a causal source (i.e. a point of origin, which may well be the endpoint from a larger perspective) can be linked to a number of downstream (cause-to-effect direction) events, properties and states. Note that this need not be linear, i.e. the direction can be cyclic. This will be explained in the main text shortly.

Two types of emergence can be defined at this point, namely *weak* and *strong* (O'Connor & Wong, 2002). Weak emergence describes the behaviour arising from micro to macrostates resultant of the interactions on the microscopic level. Note that a complete knowledge (if such a thing is possible to achieve) of component properties is not equivalent to a complete knowledge of simultaneous and collective component interactions. In strong emergence, however, the behaviour of the whole can influence the behaviour of the component parts in a form of feedback, i.e. there is a top-down causal flow.

Critically, strong emergence differs from the weak form in that the causal flow is "genuinely separate" from the influence of the component parts and is therefore said to be irreducible with respect to those components (Bedau, 1997; Chalmers, 2002). In essence this irreducibility is similar to Broad's notion of non-deducibility – the behaviour of the whole cannot be explained by the behaviour of the parts. In the arguments presented below, as well as the rest of this thesis, it will be maintained that though unexpected

behaviours can arise, and can subsequently be labelled "emergent" (depending on its definition), a depth of knowledge of component properties as well as their interactions taken together, acting simultaneously, in the context of whole can lead to the development of a framework in which such behaviour, inclusive of the top-down variety described above, can be predicted.

Complex systems and emergent phenomena are said to arise from both bottom-up and top-down collective behaviours (Johnson, 2001). The top-down perspective, or *systems view*, is a globally directed phenomenon in contrast with the purely bottom-up situation, the so-called *reductionist view* (Bar-Yam, 2004). Though behaviours arise from the short-range interactions of the component parts, e.g. short-range communication of cells in the tumour system, which embodies the concept of weak emergence, the concept of strong emergence dictates that some behaviour affects the component parts from a global perspective. The component's behaviour in turn globally affects the system by feedback. As an example of an apparent strongly emergent behaviour take tumour growth (as well as normal tissue growth in this case) that can be constrained by the global pressure exerted by its own outer cells, e.g. the viable layer of a MTS. This pressure constraint (which is a global state, i.e. top-down, induced by the original microscopic system state itself) will manifest itself on the local scale such that each tumour cell is compressed into a smaller space, which in turn forces cells to slip into the G_0 (quiescent) phase of the cell cycle – applied globally, this results in a retardation of tumour growth. Top-down control in this case has revealed itself ultimately as a combination of local interactions (the crowding affect of cells) and cell properties (induction of G_0) – importantly the combination is a simultaneous behaviour across the entire system in conjunction with other system behaviours. It is from here that one can trace the communication of behaviour through the scale and structural hierarchy of the system, from tumour (system) level down to cell level and eventually the molecular dynamics that initiate rest phase, and the chain of causality continues back up the hierarchy to affect the system globally again. Note that the property of space constraint-based retardation cannot be predicted from analysis of the state-space dynamics of the cell or molecular agents alone nor would it be deducible if it were placed in a different whole. Additionally it cannot be predicted solely from interactions of agents. It is only when the

cell interactions and properties are considered within the context of other cells, i.e. local interaction, in a systemic view, i.e. in the wider context of the tumour, does this property reveal itself.

Thus one must consider the nature of communication of such behaviour, i.e. how top-down behaviour is defined on the smaller scales of the system. From the above example it can be argued that top-down "control", if it can indeed be defined as such, manifests itself as short-range interactions between and within components and its effects therefore become cyclically bottom-up – both system and reductionist views in this case are simultaneously correct[4] because causality can flow in both directions. Non-simultaneous bidirectional causality need not be paradoxical when viewed in the language of the microcosm – top-down behaviour, the prime symptom of strong emergence, is merely the tail end of a causal chain, which is microscopically weak (in the sense of weak emergence) in origin, and temporarily continues via the global system and terminates back at the local level cyclically. This reduces strong emergence to the weak form.

When viewed in such a way it can be seen that the original notion of bottom-up system complexity (and ensuing causality) is not violated by the relatively transient causality transposed by global-local communication. The implication here is that for closed systems, where complex behaviour is independent of environmental states, any apparent top-down causality of the system is indeed bottom-up in origin. In the case of biological systems, where the systems are very much open to environmental conditions, only a subset of all behaviour is attributable to the small-scale and overall global dynamics is due to both bottom-up causality and environmental interaction-induced top-down causality.

Definition 3.11: Environment

Corollary to Definition 3.6. The environment itself can be considered a system (be it complex or simple) in its own right if the function of the system of interest is not entirely dependant on the functionality of the environment. Of course this distinction of

[4] Although there is no reason to suggest that reductionism and "systemism" are mutually exclusive. Indeed, from the reducible systems point of view, systemism actually subsumes reductionism.

dependency is a matter of degree; a precise partition between system and environment, in the case of modelling, is defined by the perspective of the modeller. Naturally, the environment represents a system(s) that is larger or equal (in physical space) to the system of interest. The environment therefore typically interacts with the system of interest in a top-down fashion (whereas the reciprocation is of course bottom-up).

Take, for example, de Pillis & Radunskaya's explanation of Jeff's phenomenon in §2.2.3.2 wherein oscillatory growth dynamics is observed as a result of immune cell interactions *in vivo*. Such oscillations can affect the distribution of nutrients, cell cycle dynamics, mutation and adaptation rates of tumour cells, oxic-level population dynamics, etc., and can therefore even influence vascularisation and further disease progression. One can assert that this oscillatory behaviour, and it's downstream effects on growth and tumour dynamics, possesses genuine causal power that cannot be deduced from examination of the components, which fulfils the basic requirements for strong emergence (Bedau, 1997). Such an assertion is, however, superficial at best – the effects of the immune system is *independent* of the tumour (Definition 3.12) when considered as an environmental interaction. Critically, the interplay of these environmental systems and their integration with the tumour give rise to complex behaviours that can be considered *weak* emergence when their interaction is viewed to form an integrated whole. In other words, when one considers all of these interactions within the context of the larger system, in this case the human body, then the origin of all behaviour can be traced back to fundamental physical laws that govern the components and simultaneously the rules that govern the *interaction between* components – again, *interaction* and properties together are the key to complex behaviour. In this case, the components of the immune system, e.g. macrophages, and their interactions with the tumour cells give rise to the aforementioned oscillatory behaviour on the macroscopic scale – the immune and tumour systems as an integrated whole have given rise to the behaviour and there is no genuine top-down causality.

Definition 3.12 System Independence

Systems A and B can be considered independent iff:

1. *There exists a subset of distinct functions f_a and f_b, independent of the superset of functions that exist as a result of A/B interactions, where $f_a \cup f_b = \varnothing$, that are carried out by A and B respectively; Or:*
2. *A and B are totally functionally independent of each other – either A or B, or both, can exist and function in the absence of the other.*

Note that system independence is a matter of degree – condition 1 is a relaxed form of condition 2 to account for the case of naturally interacting systems. Also note that condition 1 reinforces the definition of the environment with respect to the system of interest. Condition 2 does not enforce any stipulation on behavioural dynamics when A and B are together or apart. This is because the behaviour of each system may well be very different when interacting with each other, which could be the case in the naturally occurring case, e.g. tumour and immune systems, though the systems can theoretically be thought of as independent. In such cases one can consider one system as environmental (in the case of the above example, the immune system). Modularisation of systems by this criterion of independence, e.g. modularisation of all human systems, implies that complex systems can be viewed as networks of interconnected modules of functional and/or component interactions. A modular approach to complex systems is in fact fundamental to understanding complex behaviour and modelling of such systems, which will be discussed in more detail in the next chapter. System independence also raises the issue of system and component boundaries. In conditions 1 and 2 the boundaries are abstract (functional) but can also include physical aspects. The issue of boundaries is left to case-by-case specifications in Chapter 7 when system independence with respect to model integration is investigated.

Systems in general, both natural and artificial, exhibit dynamic behaviour. To delineate a distinction from so-called chaotic systems where state dynamics are also in non-linear flux, albeit in an extremely disorganised fashion, behaviour in complex systems is more organised, structured and hierarchical as described above[5]. To define this distinction

[5] Self-organisation (described later) is another behaviour that sets CS apart from chaotic systems – organisational ability implies a reduction in entropy of the whole system. Indeed living systems have been shown to be able to dynamically change their Gibbs free energy potential (defined as the energy available to

further, complex system behaviours are relatively insensitive to initial conditions and eventually stabilise, whereas chaotic system behaviours vary hugely even with small initial state changes (Bar-Yam, 1997; Wolfram, 1988). This convergence of behaviour from differing initial conditions, and indeed the resilience of CS to external stimuli, can be viewed as a type of adaptation, e.g. the tumour system has the adaptive ability to dampen sensitivities to adverse conditions (Awale *et al.*, 2006). Tumour systems have therefore been described as complex adaptive systems (CAS) with respect to self-sustenance and therapy resistance (Pienta, 2006) though the property of adaptation is far more fundamental than this – supporting cellular systems in a dynamic environment itself is an extremely complex adaptive behaviour. The induction of angiogenesis by over-expression of TAFs and avoidance of apoptosis, for example, can be viewed as adaptive behaviours in a highly dynamic environment. Most often the mechanism of adaptation in the case of tumours is an accelerated mutation rate (Natarajan *et al.*, 2003) where a selection pressure, such as application of therapy, together with a genetically diverse cell population can give rise to therapy-resistant cells. The tumour system then replenishes with the resistant cell variety and disease persists by this Darwinian process, where one can identify individual cells as mutual competitors. There are a number of sensitivities for which the tumour system cannot adapt too such as susceptibility to rapid mutation leading to unviable cells and, to take an extreme example, death of the host. Subsystem controls and microscopic dependencies are responsible for this "robust but fragile" behaviour (Csete & Doyle, 2002), which has been described for natural and artificial systems alike.

Another property of complex systems is self-organisation (Coffey, 1998), e.g. the organisation of the cytoskeleton and spatiotemporal organisation of molecules participating in signal transduction cascades within a cell. Self-organisation on the macroscopic level is defined to be the result of structural organisation on the meso/microscopic level, which in turn is due to local interaction between agents (Coveney & Fowler, 2005). The properties of self-organisation and adaptation due to the interactions of agents are perhaps more evident in biological systems than any other set of complex

---

carry out any process, e.g. synthesis, taxis, etc.) to push the system into decreased entropic states (Boogerd *et al.*, 2006).

systems. Therefore the recent pattern, as described in the previous chapter, of some modellers moving away from traditional mathematical techniques in favour of interaction/rule-based techniques should not be so surprising since a concept such as self-organisation is very difficult to describe mathematically without trivialising mechanistic behaviours[6]. In the context of tumour systems self-organisation becomes an interesting problem since it is the very loss of normal organisational control that causes over-proliferation – the normally robust tissue organisational behaviour, tightly controlled at both the cellular and molecular levels, reveals its fragility to induced and/or spontaneous mutations that escape apoptosis and early immune detection.

In summary complex systems have been discussed in the context of their properties, the most notable being that of emergence, nonlinear behaviour and self-organisation. Critically it has been argued that it is the interactions between agents within the system and across the environment, as well as their properties, when considered simultaneously in the context of whole (i.e. all of these interactions and properties *together*) that are responsible for overall observable behaviour. Moreover an explanation of systems on multiple physical and time scales is entirely conducive to an interactions-centric understanding of complexity that goes hand-in-hand with properties-centric understanding on the microscale. Importantly such an understanding holds the key to model integration strategies, which will be explained in the next chapter.

3.4 A Systems-Theoretic Approach in Biology: "System Biology"

The volume of research pertaining to simulations of biological phenomena, including cancer, has accelerated dramatically in recent years with high performance computing, e.g. GRID technology, and the availability of enormous amounts of molecular data via high-throughput technologies such as microarrays (Brazma *et al.*, 2001) has generated a surge of interest in understanding biological systems quantitatively on the minute scale. The field of systems biology embodies a sizeable portion of this research. As the name suggests, the field in principle spans entire cell-, tissue-, organ-, organism- and

[6] Moreover note that biological systems can exhibit a dynamic structure of organisation, not just a rigid one.

community-level understanding via modelling, simulations and visualisations though, as will be explained below, the main focus has centred on intracellular pathways. Systems biology is therefore a fusion of systems theory and mathematics with computer science applied to biology and readily lends itself to the inclusion of engineering principles. In addition to the systems perspective the eventual use of systems biology is closely linked, but not equivalent, to bioinformatics. Systems biology is not as new a field as one might be led to think – questions as to the relevance and application of systems theory to biology were seriously being considered as early as the 1950s (Cornish-Bowden, 2005), though the uptake by the larger community was still rather reductionist in spirit, e.g. elucidation of vast metabolic pathways of mammalian and plant cells, and one could argue that it is as old as systems theory itself (in fact Ludwig von Bertalanffy, the founder of systems science in the early 20th century, was himself a biologist by profession). Indeed systems thinking has often gone hand-in-hand with biology though applications have not directly been biological, e.g. John von Neumann's cellular automata theory in the 1940s was described in the context of reproductive artificial life (hence Conway's Game of Life in 1970) and predator-prey equations developed by Lotka and Volterra in the mid 1920s which still play an important role in describing both ecology and cellular competition in many current modelling schemes.

Currently systems biology is shifting deeper into systems thinking such that it is paving the way to ever more complex applications of modelling and simulations to solve biological problems, and conversely the understanding of complex biological systems is providing a drive in computer science, mathematics and engineering to improve on current technology and methods. These two channels embody subfields that can be described as computational biology and biological computing respectively (Giavitto & Godin, 2002).

Wolkenhauer *et al.* (2003) describes the key challenges for systems biology as follows:

- Methodologies for parameter estimation.
    - As can be seen from the heavy reliance on differential equation and rule-based approaches in the tumour modelling literature, parameterisation is an

important issue for accurate simulations. This can only be obtained from wet-lab experimentation and data mining. As will be explained in following chapters, the model integration process can also drive the search for parameter values.

- Modular representations and simulation of large-scale system dynamics.
- Scaling models across time-scales and description levels (genes, transcripts, proteins, etc.)
    - Model integration is recognised as a key technology in systems biology (Finkelstein *et al.*, 2004). Here it is defined further as a technology that addresses the scale and model semantics problem too. Simulator software integration is one of many integration problems tackled by Systems Biology Workbench[7] (SBW) (Sauro *et al.*, 2003) and SCIpath (Patel & Nagl, 2004), as well as other middleware.
- Visualisation and fusion of information, integration of models and simulators.
    - Visualisation is a critically important aspect of systems science (Hnat & Chapman, 2000; Drew & Hendley, 1995). Visualisation is pivotal in turning complex data and computation into human understanding.

Fundamentally, amongst the majority of currently active systems biologists, the bulleted points above distil to a single purpose – to understand the mechanisms of regulation, and therefore the dynamics, of biological systems (Wolkenhauer *et al.*, 2004) via enabling technologies and visualisations. Ultimately this involves modelling of the interplay between environment and microscopic local interactions of cells, molecules and other entities of interest, and the system of interest on the mesoscopic scale. Perhaps due to the huge efforts put into genome projects and greater understanding of molecular structure and events via improved technologies, the original systemic focus has reduced for many researchers to regulation of the genome. It is thus little wonder that the central topic that has dominated systems biology for the last eight years is molecular networks (metabolic and signal transduction), even though the field portrays itself as a richly varied mixture of perspectives as shown by the quotes at the beginning of this chapter. Mathematical

[7] http://www.sbw-sbml.org/

modelling and simulations of pathways have received a great deal of attention from a computer science perspective (Sauro *et al.*, 2003) whereas the topology of networks (Adiwijaya *et al.*, 2006), modes and (de)centralisation of control (Barabasi & Oltvai, 2004) and robustness (Kitano, 2004) have been tackled from an engineering perspective. With a plethora of pathway modelling devices and databases, this trend of research is set to continue.

Given the current pathways-oriented focus in systems biology it would seem that the presumed goal is a fuller understanding of the cell, rather than a fuller understanding of tissue, organ or whole-organism systems (though it may well be the case that a hierarchical understanding, starting from the cell level, is sought). The academic question in this regard is whether pathway dynamics really *should* embody the bulk of systems biology research. Is it not the case that aiming to understand the component agents (pathways) to explain the behaviour of the cell system is reductionism by definition? Given the previous discussion on complex systems the answer is affirmative if pathways are considered isolated components of the whole, i.e. taken out from the context of the system. Unless this context is considered, the determination of pathway dynamics, whilst extremely powerful in its own right, is essentially an attempt to describe biological systems *serially*[8] and purely bottom-up and is therefore contrary to the view of systems thinking (i.e. consideration of properties and interactions of components simultaneously in the system). Indeed, as can be seen from §2.2.4, the vast majority of pathway analyses and models are not placed back into systemic context, i.e. system dynamics is not considered. It is not implied here that such modelling is invalid – indeed pathways are components of the whole (cell) and serial additions make knowledge of that subsystem of the cell more complete. However, in the opinion of the author, whether this will complete knowledge of cellular behaviour as a whole is an open question. The cell is considered a complex system its own right; its behavioural dependencies in environmental conditions raises the question of whether an isolationist view of pathway dynamics will ever really reveal cell behaviour to the point of tissue-, organ- or organism-level understanding (as may well be required for

[8] Pathways in the real system are incredibly complex and so the intuitive way to build up a systemic picture is to break them down into smaller pathways that are examined in isolation and later joined serially by common nodes.

cancer systems, for example). Encouragingly, developments such as the Physiome Project (discussed shortly) seem to guiding this research into a more systemic view.

This pathways-oriented view has been provided by the strong influence of engineering principles within the field – Hiroaki Kitano (System Biology Institute, Japan), John Doyle (Caltech, USA), Uri Alon (Weizmann Institute of Science, Israel) and Douglas Kell (University of Manchester, UK) who have all come from engineering-oriented fields, and groups such as ERATO (Japan) that produced Systems Biology Mark-up Language (SBML) with Caltech computer scientists, have made extremely influential contributions to the field via literature and software that have spurred on the continued research in pathways. Of course from the engineering perspective, where similar pathway principles have been drawn from artificial systems, the understanding of pathways *is* systems-oriented – networks of interacting individuals form the basis for overall behaviour as explained earlier but when applied to biological systems this has been capped to the cell level.

Cell-level research is of course extremely useful in its own right, as recent developments have shown. Genome sequencing, two-hybrid and microarray experimentation, all play an important role in driving pathways research to progress the understanding of cellular circuitry (Kitano, 2002). However, assuming the experimental data and thence the models and simulations are correct, to what extent will the understanding of these complex dynamics contribute to understanding the behaviour of the cell as a community member of an *in vivo* tissue? This is as yet an open question since the coupling of pathway dynamics with cellular behaviour is not always clear – pathway dynamics itself (i.e. flux of molecular constituents) is the primary output but the collective profile of that output is often difficult to relate to cellular behaviour. The danger is that the "systems" biology viewpoint has focused on the cell system as the proverbial whole and has treated either molecules as agents, wherein molecules are linked to other molecules to form more complex pathways and networks, or pathways as agents, wherein pathways are linked to other pathways in an additive fashion. The behavioural dynamics of pathways, and ultimately cells, have therefore been reduced with the exclusion of other cell behaviours.

The previous chapter presented projects modelling pathways in isolation from the rest of the potential, but unknown, network linkages (and linkages to other stimuli such as molecular-level changes in the membrane due to physical stress and strain), i.e. there is a real danger of developing 'pseudo-networks' due to incomplete knowledge. When one thinks about as yet undetected interactions, which are omitted by necessity from the models, serious questions are raised as to how well these models actually represent reality. In summary, though systems biology promises a solution to understanding the complexity of life it is still a field with many serious challenges.

There is, however, a very real sense of systemic-oriented change within the field. In fact many systems biologists have turned to the wider context of the whole with software infrastructures, such as E-Cell and Virtual Cell[9], and global ventures such as the Physiome Project, of which their have been offshoot projects, notably the Cardiome, Epithiliome and Beacon Projects (discussed later). One can see that the focus is changing from a cellular perspective to a systems-wide perspective with recognition of the fact that the cell, a complex system unto itself, is only a small component of much larger systems and that to gain any real insight, especially for clinical use, the mesoscopic scale (cellular focus) must be expanded towards tissue and organ levels. Tissues and organs are of course themselves components of the whole organism. Obviously the importance of complex agent and component functions cannot be overstated – the molecular level dynamics in the cell, for example, is vital in understanding overall behaviour and so even if the shift in focus is towards a larger (physical and/or time) scale the smaller scale dynamics must not be ignored. This is a very important challenge in modelling biological systems – building cooperative models at different scales that constitute the whole, wherein the inherent layered hierarchy of complexity of the overall system is addressed, is highly nontrivial. The role of model integration is to play a critical part in the solution to this problem, as will be explained in the next chapter.

[9] Note that though the original focus of these projects were centred on the cell level they are moving to the tissue level with the availability of more computational power.

3.5 Coping with Complexity: Modelling of Complex Systems

It is clear from the previous sections that an interaction-centric approach to complex systems is key to understanding the diversity of behaviour and this is especially so in the biological case. This raises the next question of how can any system behaviour be quantified formally? Systems can be viewed in terms of states and state dependencies, which are the prerequisite to understanding behavioural dynamics as seen in the previous chapter. To formalise further:

- States, or *observables*, are described in terms of *state variables* (e.g. nutrient concentrations, diffusion, proliferation rates, etc.).
- States can change over time according to a *transition function* (Equation 3.2). The transition function itself is sometimes subject to change, and therefore one can obtain several orders of transition function for a particular system[10]. In practice this is sometimes difficult to conceptualise but some formalisations do exist, such as finite state machines.
- The set of all possible states is termed the *phase* or *state space*, which is not always possible to define explicitly.
- The transitions of states over many time indexes can be recorded as a *trajectory*.

There are many techniques by which one can begin to model complex systems given basic mathematical and computational tools. Some of these are discussed below and form the basis, either directly or indirectly, of many of the modelling efforts summarised in the previous chapter.

When viewed as observables states are seldom elemental. This means that other states of the system, which may not be directly measurable, collectively contribute to the value of the state variable. In that case it is conceivable that there exists a number of sub-state combinations that yield identical phenomenological state (observable) values. A complex system can then be modelled abstractly as a hierarchical tree with state and sub-state

[10] Though systems are rarely modelled with something more complex than a 3rd order function.

relationships (becoming more atomic as one reaches the termini of the tree) including combinatorial identities. Wolkenhauer (2001) and Bar Yam (2004) both ascribe to this phenomenological view of the system wherein only the observables are taken into account. The rationale is that the more atomic (given some modeller-defined cut-off) state dependencies and dynamics need not be modelled explicitly, thereby saving on computational cost, since collective observables can be used reasonably to explain behaviour. A simple formalisation is as follows:

$$\psi_t : \{s_{t,1}, s_{t,2} \ldots s_{t,y}\} \qquad \text{Eq. 3.1}$$

yields the definition of a system given y observables of interest where:

ψ_t is the overall system state at time t, consisting of a set of observables.
$s_{t,x}$ is the x^{th} state (observable) variable at time t.

Every state undergoes change by some state transition function:

$$s_{t,i} \xrightarrow{f_s} s_{t+1,i} \qquad \text{Eq. 3.2}$$

The transition function, f_s, can be more complex, involving multiple time steps, states and environmental inputs. Additionally, if more detail is required, the model can be extended by considering collectives of states that contribute to more phenomenological states. When considering state variables as discrete one can view the transition of states via the state function as a model of a finite state machine (by cloning the transition function for every state-to-state transition) – in fact a state machine may well be the starting point in a simple modelling exercise. Superficially the transition function's relation to the state(s), along with ψ_t as a whole, constitutes the definition of system behaviour (see Definition 3.2). Whilst this will suffice as a rudimentary model, one can see immediately that it is no more in substance than the phenomenological models described in the previous chapter. The mechanistic basis of behaviour is hidden within the transition function. Whilst there are

very obvious advantages to this methodology, as will be seen in the explanation of metamodelling approaches in §4.3.1.3, there are also distinct disadvantages. For example, in the case of tumour models, research questions will often revolve around the mechanistic basis of behaviour since this knowledge can greatly benefit how therapy can be developed and improved: what are the molecular explanations of cellular behaviour (e.g. pathway dynamics that contribute apoptotic decision-making)? What are factors that affect the spatiotemporal development of neovasculature? Why does growth retardation increase even if there are enough nutrients as the tumour grows? These are just a few questions that can be raised. In short, if more mechanistic explanations of behaviour are needed for detailed analysis then this phenomenological approach is insufficient.

3.5.1 Individual-based Modelling Methods

As can be seen from the literature review in Chapter 2, individual-based models such as CA and ABM have gained more popularity as viable methods to cope with the complexity of biology. The relative simplicity and flexibility of such methods coupled with readily available software, efficiency, amenability to parallelisation and superiority over traditional techniques as explained earlier makes individual-based modelling an attractive alternative to cope with biological complexity (Alber *et al.*, 2002). Individual-based techniques shift the focus of modelling from a phenomenological to a local interaction-centric perspective (Bernier *et al.*, 2002). In the case of bio-simulations with ABMs, for example, the bulk of the modelling effort is focussed on the agent, e.g. cell, and its interactions and behaviour rather than global dynamics of the tissue or organ. The volume of related modelling literature is enormous and only a small fraction with particular relevance to the work presented here is explained in any detail.

CA were first developed by one of the greatest mathematicians of the 20th century, Professor Jon von Neumann and his contemporary Stanislas Ulam (Wolfram, 2002). Since then CA have been used and developed in many applications that include quantum mechanics, diffusion, hydrodynamics and engineering, general systems theory, cryptography, game theory, economics and biology (Bar Yam, 1997). The concept of the

cellular automaton is extremely simple and is illustrated in Figure 3.1 in its simplest form – a 1-dimensional CA.

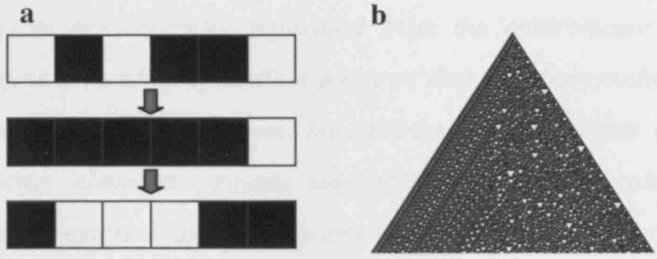
The deterministic transition rules determine the next state of a cell[11], which is totally dependant on the present state of the cell itself (with some variations that include a concept of limited cell memory) and the present states of neighbouring cells, i.e. cells only behave according to their local neighbourhood. The direct relevance to biological systems here is obvious wherein biological components such as living cells behave according to their own states as well as the environmental state(s) – the one major difference here is that the real system is continuous in nature whereas a CA is discrete both in space and time. It is easy to see how this can be transported into higher dimensions (of course only up to three for real-world simulations). Stochastic approaches have been incorporated into CA, e.g. Qi *et al.* (1993) use stochastic rules for cellular interactions in a growth model. According to the transition rules and configuration the resulting behaviour is observed to be very complex, repetitive or pseudo-random (Mizraji, 2004), however this kind of distinction of behaviour has only ever been examined rigorously with 1D binary state CA. Stephan Wolfram, who greatly popularised the utilisation of CA in the 1980s and 1990s, characterised the complexity of behaviour of 1D binary state CA but failed to describe their dynamics with traditional mathematical methods, leading him to propose that complex systems dynamics therefore cannot be described purely by traditional mathematical methods (Wolfram, 1988).

Many different variations of the basic CA described in Figure 3.1 exist. Rules can be complicated but are generally kept quite simple in most applications. Two of the most popular CA that are used routinely in systems science and hydrodynamics are the so-called Lattice Gas (LG) and Lattice Boltzmann (LB) flavours, which incorporate important physical laws into the rules, e.g. Newtonian mechanics[12].

---

[11] Note use of terminology: the term "cell" is used for the individual in CA as well as the biological type.

[12] The associated mathematics has in recent years become extremely well defined but quite extensive and complex and so will not be discussed here specifically – interested readers are directed to Succi *et al.* (2002) and Vesely (2001).



Current State	BBB	BBW	BWB	BWW	WBB	WBW	WWB	WWW
Next State	W	W	W	B	B	B	B	W

Figure 3.1 a. Discrete blocks (cell, or finite state automaton) make up a single line (i.e. 1 dimension). This initial state is called the *configuration*. The table above is called the rule-base, which explicates the rules that each cell must follow at each discrete time step. So if a cell is Black, and the two cells to its left and right are black, then its next state is White (the first rule in the table). This particular rule-base is defined as Rule 30 by Wolfram, which comes from turning the logical rules into binary (not discussed any further here). The rules operate on a *neighbourhood* of radius 1, i.e. they only observe the state of cells "1-cell away", however many different types of neighbourhood have been used (the most common being the von Neumann and Moore neighbourhoods, which are diamond- and dice-shaped respectively). The singular sequence is usually considered toroidal (i.e. left side is joined to the right side to make a ring. A donut, or toroid, shape is made if the CA is 2D). **b**. Rule 30 exhibits a complex pattern. Notice that one can observe tiny triangles emerging from the simple rules – these can be interpreted as fractals. Image adapted from Wolfram (1994).

CA cells can be considered as a type of static (in space) agent, as per Definition 3.8, and so CA can be viewed as a special case of agent-based systems (ABS). Table 3.1 illustrates the fundamental similarities and differences between CA and ABS.

Definition 3.13 Computational Agent

(Corollary to Definition 3.8) A computational agent can be defined as an autonomous, or at the very least semi-autonomous, entity within a software system and is designed to perform some function based on predefined logical rules (Giraldi, 2005). Agents are often encoded with some degree of intelligence to achieve local or global goals but communication with other agents is key to agent functionality (communication with the parental system is sometimes also just as important). States (usually discrete) can be embedded into agents, which can then have downstream effects on behaviour via a memory or learning – they can therefore also be viewed as state machines. Agents have been very successful in many application areas from optimisation and mathematical

solving tasks to simulation avatars and industry-standard AI deployment (Van Dyke Parunak, 1999). Agents are logically separated from the environment in which they operate. One can define a number of distinct features that are discernable in most ABS: Precepts are the information flows between the environment (e.g. other agents) and the agent; actuator/effector elements provide an information-sending ability by which precepts can be sent to exterior agents; sensors are the precept-sensing faculty of an agent; processing elements make up the internal anatomy of an agent and is highly dependant on the overall ABS architecture as well as the intended functionality of the system – it may well include memory and learning faculties for intelligence capability, for example.

	ABS	**CA**
Interactor	$Agent = \alpha$	$Cell = \chi$
Continuity	*Discrete/Continuous*	*Discrete*
Position	$xyz \begin{cases} = f(\Delta t) & \text{if spatial} \\ \neq f(\Delta t) & \text{otherwise} \end{cases}$	$xyz \neq f(\Delta t)$
Environment	$E = Env(\alpha) \leftrightarrow \{e_{1\ldots n}\}$	$E = Neighbourhood(\chi) \leftrightarrow \{e_{1\ldots n}\}$
State	$S = \{s_{1\ldots m}\}$	$S = \{s_{1\ldots m}\}$
Rule Base	$R = \{r_{1\ldots p}\}$ $Action : d(R,S,E)$	$R = \{r_{1\ldots p}\}$ $Action : \mu(R,S,E)$
State Transition	$S_{t+\delta} \wedge E_{t+\delta} : Action_t$	$S_{t+\delta} : Action_t$

Table 3.1. Similarities and differences between ABS and CA. These are only very abstract descriptions; there exists a plethora of variants and hybrids. Above, *d* denotes a decision function, i.e. intelligent selection of appropriate action, which in the special case reduces to a matching function μ that uses a simple state-lookup to perform the appropriate action; *Env* and *Neighbourhood* represent local interactor-capture functions (i.e. a definition of locality); $\delta \rightarrow 0$ in the continuous case. All other symbols are as in main text (refer to the next chapter for a more in-depth description of the significance of the symbols used here).

ABS can be extremely varied with types focussing on differences in the way agents behave, taking into account differences in "anatomy" (sensors, actuators, etc.), goals (function) and rule base types, to the environments (contexts) that they interact in and inhabit. In the case of this thesis the type of particular interest are those that exist within

simulation systems, i.e. the computational analogue of the agents discussed earlier with reference to complex systems. These agents are different from the kind that one, for example, might encounter in optimisation methods where the agent is goal-oriented and perhaps communicative under a specialised architecture. The simulation system, in the case of real-world simulations, would constitute a computational representation of a limited 3D space in which components, i.e. agents, interact according to preset logical rules (note that rules do not have to be static – they can be learned, for example) and states. Presumably most agents would represent real-world objects and would therefore need to behave in a similar way in terms of state, space and time.

Of course there is always precedent for more complex abstractions. Take the typical case of a large tumour that can contain millions of tumour and normal cells (Ribba *et al.*, 2006), and would be too difficult to model on a strict individual basis – a more abstract approach will see each agent within a coarser-grained simulation representing a number of cells acting collectively instead. Hybrid systems – mixtures of the individual-based ideas described here – are also commonplace and provide a powerful means by which the most desirable parts each particular mechanism can be utilised. The model integration solution described in the next chapter is such a hybrid system where CA and ABS are combined to form a representation of the real-world complex system of interest.

3.5.2 Simulation Software

Formalising a platform for simulations of complex systems is a nontrivial task. Simulations of complex systems provide the key to future experimentation, hypothesis generation and intelligent prediction and hence such a formalisation is very important.

The engineering field in particular has taken the modelling and simulation of complex systems seriously and a number of platforms now exist[13], many of which are focussed on one or a particular class of system (Wolkenhauer, 2001). Most platforms are commercial. Many applications also work complimentarily to other popular packages such as

[13] A comprehensive list of generalised software can be found at http://www.idsia.ch/~andrea/simtools.html

Mathematica[14] and MATLAB. A number of platforms make use of parallel computing, such as the Parsec system (Bagrodia *et al.*, 1998). Other packages include ColSim[15] and the Discrete Event Simulation (DEVS) formalism, upon which some older simulation frameworks are based (Zeigler, 1993). None of these however has been specifically created for the simulation of biological systems.

Two relatively new packages are Modelica[16] and Ptolemy II[17], which have already proved to be quite popular. Modelica is quickly gaining ground in becoming a new standard in the open source market as a viable alternative to commercial simulation and mathematical packages such as Mathwork's Simulink. Modelica actually constitutes two layered components that are continually being improved and built on: (i) a field-independent model representation language, and (ii) a software infrastructure that runs models written in that language. Many applications have sprouted from this effort including complex aeronautical simulations and other "multi-physics systems" (Otter & Elmqvist, 2001). Modelica is of particular interest since it explicitly makes use of model/sub-model relationships and facilitates model-building in this way to promote the idea of model reuse. A GUI can be used to create specific modelling components, which can then be integrated with other models using a specifically defined mathematical relationship. This is supported by the idea of partial model generation, which allows the user to create modelling components with missing mathematical relationships that can be later realised (i.e. integrated with another partial model). As will be explained in the next chapter, this idea constitutes a simple linear integration akin to many previous model integration schemes (Geoffrion, 1998; Dolk, 2001). However there are limitations – so far the software only supports model integration with a restricted set of mathematical tools, namely differential, algebraic and discrete equations. The creators of Ptolemy II, which works along a similar principle using an OOP-like architecture to build and improve models, have made a concerted effort to include a wide variety of models that can be used within the system including finite state machines, event-driven simulations (discrete event)

[14] http://www.wolfram.com/products/mathematica/index.html
[15] http://www.colsim.org/
[16] http://www.modelica.org/
[17] http://ptolemy.eecs.berkeley.edu/ptolemyII/

and a number of hybrids (Bhattacharya *et al.*, 2005). However, an appreciable amount of recoding is required if a new model type that cannot be reconstructed using the basic component types in the standard library is to be integrated into the system and represented in Ptolemy's model language (MoML). Additionally both Modelica and Ptolemy II have been specifically created for (engineering-oriented) physical systems rather than complex systems though the creators claim that given enough computational power the software can cope with massive amounts of calculations efficiently (hundreds of thousands of equations in the case of Modelica, which is in the same order of equations that was used for the Cardiome Project described below).

Two of the most prominent efforts in complex systems simulation on the cellular level are the E-Cell and Virtual Cell projects. E-Cell was one of the first serious efforts to materialise from systems biology in 1996 attempting to simulate the behaviour of a single cell. Takahashi *et al.* (2004) describe recent efforts in the E-Cell project, including the multiscale (time and space) integration of submodules to build up the simulation. The group report some success with the integration of relatively simple implementations of ODE, Gillespie and ODE/Gillespie composite models. They are still actively working on different modules of the cellular system – for example, a model of the mitochondrion has been recently reported by Yugi & Tomita (2004) – as well as moving into microbial simulations.

The Virtual Cell project is a comparatively new venture with the same ultimate goal of whole cell simulation. Simulations have a heavy reliance on the description of interactions that are controlled by defined differential equations (communicated in Virtual Cell Mathematics Description Language), and the package is equipped with numerical solvers as well as a user-friendly interface (Loew & Schaff, 2001). Virtual Cell explicitly accommodates structural information (though non-spatial modelling is also possible), which is not as apparent in E-Cell, and is based on an ontological geometry framework. The framework defines the morphology of the cell upon which the inputted mathematical models operate. The morphology throughout the simulation, however, is restricted to being completely static whilst the spatiotemporal allocation of molecular networks within

the cell (e.g. pulse of current through a neuron) is dynamic. Simulations can run in three dimensions and provide quantitative results within the formalised framework. The simulation engine and database is centralised at a high-performance server, making it available via the web, and implemented in Java (Slepchenko *et al.*, 2003).

There are many other cell-based simulations including the CyberCell Project[18], which is focusing on modelling of *E.Coli* cells in a purely bottom-up agent-based approach. Discrete agents, i.e. molecules, are simulated to obey biophysical laws so as to mimic cellular pathways. Of course this method is very computationally expensive. Others include MCell[19], a Monte Carlo modelling-based platform for microphysiology, and a number of efforts coordinated by the International *E.Coli* Association (IECA) (Holden, 2002), WebCell (Lee *et al.*, 2006) and various commercial ventures, e.g. Genomatica[20] and Physiomics Plc[21], who also attempt more macroscale simulations. A relatively recent development is that of the Silicon Cell group (Snoep *et al.*, 2005), powered by JWS Online[22], which is a data-driven approach to pathway modelling.

The Physiome Project is currently the most well known biologically related complex systems simulation project with an impressive simulation of the heart, its flagship project, already completed (Hunter, 2004), though improvements are ongoing. This project is an international collaborative effort that includes similar biological simulations for the lung, musculo-skeletal system, digestive system and liver. Interestingly an e-Science project has started quite recently headed by David Gavaghan[23] at Oxford University. The vision of the Integrative Biology[24] group is the initiation of a Grid-based platform on which multiscale simulations can be performed, including visualisation and data sharing on a global collaborative scale. Latest reports[25] indicate good infrastructure progress, however the platform itself is not yet ready for public release.

---

[18] http://www.projectcybercell.com/
[19] http://www.mcell.cnl.salk.edu/
[20] http://www.genomatica.com/index.shtml
[21] http://www.physiomics-plc.com/
[22] http://www.jjj.bio.vu.nl/index.html
[23] http://web.comlab.ox.ac.uk/oucl/research/areas/biocomp/tumour.html
[24] This is in fact an offshoot of the older Physiome Project.
[25] http://www.integrativebiology.ac.uk/progress06.html

A subsidiary of the Integrative Biology group is the Epitheliome Project[26], focuses on epithelial tissue and is still ongoing. In contrast to the arguments presented in §3.4 where it was stated that systems biology, with respect to pathways modelling, has too often been capped to the cell level, the Epitheliome Project strives to describe multi-cell behaviour and does so by crossing genetic, pathway, cell and tissue levels[27] by agent-based modelling. Interestingly, Smallwood & Holcombe (2006) describe a hierarchy of models to cope with complexity, using a metamodelling approach to describe lower-level dynamics (metamodelling will be discussed in more detail in §4.3.1.3). This work has led to the development of the Flexible Agent Modelling Environment[28] (FLAME), a parallelisable software package for modelling biological systems as well as other systems (recent developments have included modelling of the European economy).

A recent and prominent tumour simulation effort, though not formalised into any software platform like FLAME, Virtual Cell or E-Cell, is proposed by Alarcon *et al.* (2004). The model presented focuses on blood flow and heterogeneity of the tumour. This paper was, until recently, the only one in the literature to the best of the author's knowledge to describe tumour dynamics (in terms of growth) on such a wide scale, spanning from a molecular activity (including p27, cyc, cdh for control of cell cycle) to blood flow and response to chemotherapy. CViT[29] is an emerging community of systems biologists that aim to create a virtual tumour. However, to the author's knowledge, this group are still in the preliminary stages of providing an online modelling archive and have not yet reached a whole tumour simulation stage. One of the latest modelling efforts is described by Zhang *et al.* (2007), which is a cross-scale ABM (described in Chapter 7). Specific software engineering details or new modelling methodologies have not yet been disclosed.

[26] http://www.epitheliome.com/
[27] http://www.dcs.shef.ac.uk/~rod/Integrative_Systems_Biology.html
[28] http://www.flame.ac.uk/
[29] https://www.cvit.org/

3.6 Summary

Complex systems have been briefly described as systems in which there are often very large numbers of interacting agents that collectively display both linear and non-linear behaviours, as well as self-organisation. These behaviours can often be described as emergent. At least in biological systems these behaviours, in conjunction with environmental interactions, can be described in the context of local interactions of the agents, e.g. cell-to-cell/intra-cellular space interactions, and their internal states. In recognition of this fact several efforts have arisen in systems biology, such as the Physiome Project, which aim to model biology at various levels of granularity – however not all have taken this paradigm to the individual (agent) level, favouring DE-based techniques instead. A number of simulation software projects and modelling standards have arisen that describe biology at the level of signalling pathways, which has driven systems biology to its current state.

3.7 Conclusion

The vast majority of software and modelling projects are working towards a system-wide view but the extent to which such efforts have been successful is as yet arguable. The true yardstick by which success can be measured with respect to tumour modelling will be in how well models will be able to provide mechanistic explanations to complex behaviour and thereby present a viable protocol for therapy development. Many of the methodologies described above, in combination, can conceivably be used to this end however one must bear in mind the sheer complexity of the system in question. Additionally the agent-based nature of biological systems, which are very difficult to model with traditional techniques, must also be considered in serious model efforts. Whilst the system can be broken down into functional components which can often be modelled separately, these models then need to be integrated together to form a computational whole and it is this need that is the prime motivation for this project. The next chapter will discuss what this process is, how it can be done and ultimately how this knowledge can aide the systems biology field in the

search for extensive, viable, systems-level *in silico* simulations – and most pertinently to this project, what role model integration will play in realising these goals.

4. Model Integration from a Systems Perspective: Formalisations

4.1 Chapter Objectives

- To define and review model integration with respect to the current technologies.
- Based on the literature review in Chapter 2, to define models in terms of:
    - Paradigm (formalism).
    - What the model actually represents.
    - The associated semantics, e.g. model assumptions.
    - Scale.
    - Behaviour
- Based on the points above, to define the core challenges of model integration.

4.2 Brief

The previous two chapters clearly demonstrate the predictive and analytical power of mathematical and computational techniques. Many published models have demonstrated the ability to predict non-linear behaviour to a limited extent and to this degree it can be argued that these techniques can gain ground as viable substitutes in at least the preliminary steps of the research cycle as has been the case in the engineering fields. However it is vitally important to recognise the fact that models in the field of systems biology are as yet simplifications. Even the more complex models among those reviewed in Chapter 2 have been simplified to focus on a select number of processes and subsystems, limiting their use to the corresponding biological mechanisms and therefore restricting their clinical viability. To build more representative and detailed simulations one must start to integrate models together wherein the individual models represent different or overlapping components of the system.

Definition 4.1 Model (mathematical, computational and logical)

(Corollary to Definition 2.1) An abstraction of one or more observable system of entities, states and/or behaviours such that the artificial abstraction reflects reality to some predefined degree of accuracy (if such a metric is known). Models can arise in a diverse number of forms (explained shortly). Mathematical, computational and logical models use mathematical, computational and logical language respectively, or a mixture (hybrid model), to describe the system in question. Logical varieties are often categorised under the computational category since a set of logical arguments that compose such a model is often translatable into a computational one. The term model is used abstractly throughout this text to mean any of these types unless otherwise stated. A more detailed definition of models of these types is given in the main text.

Definition 4.2: Model Integration

The process by which mathematical, computational and/or logical constructs can be incorporated into a single framework such that the resultant model is representative, to some reasonable degree of accuracy, of all of the original constructs. In this case "representative" means that the resulting model integrates the behaviour and semantics, but not necessarily the precise formalisms, of the original constructs. The integrated model need not run (be solved) within the same framework. The three different views on model integration can be summarised as below (discussed in more detail in §4.4):

1. *Isolative Communication (Model Sharing): Model integration is limited to translation into a lingua franca, which can then be imported into different frameworks (Dolk & Kottemann, 1993). The responsibility of integration (with respect to types 2 and 3) and model solving is then shouldered by the importing application.*
    - *E.g. SBML (explained later in this chapter) is classified as isolative since it only describes a model mathematically, as well as related semantics, but passes off the responsibility of integration to the application(s) that import and interpret it.*
2. *Integrative Communication: Models run independently with middleware that communicates states so that models can coordinate globally. The software*

components and languages the middleware uses become the focus of model integration research.

- *E.g. Systems Biology Workbench (SBW) enables live applications to share objects, such as simulation states. This is different from isolative communication where the communication policy is static, such as an XML file.*

3. *Decomposite: Models are decomposed into component parts, or collections of components and are converted into other runnable components, and are solved under a single framework. This can often reduce to integrative communication on a finer scale, i.e. model to submodel decomposition and communication. There are various degrees of decomposition, discussed later in this chapter.*
    - *To the best of the author's knowledge only the integration strategy developed in this project comes under this category (see next chapter).*

In order to clearly identify key challenges involved in model integration strategies this chapter details the results of a detailed investigation into the nature of models using the literature review, summarised in Chapter 2 and Appendix A1. Different model integration strategies are compared and a novel agent-based strategy is proposed wherein many if not all challenges can be met.

4.3 The Nature of Models

An understanding of the very nature of models and their modes of use is critical in developing a formal platform on which integration can take place. Any model integration technology would need to be robust to the diversity of modelling techniques and so it becomes essential to develop an understanding of this diversity, from where it originates and on what levels it occurs so that:

1. The complexity of the model integration problem can be stated clearly.
2. A formal methodology for model integration can be explicitly defined.
3. Software design of a model integration platform can be defined.

4.3.1 Abstract Views of Models

It is clear that a complete low-level classification of models for the purposes of model integration is futile due to the sheer number of model types, hybrids and contexts in which models are used. Even if such a task is undertaken, by the time one could finish such classification it is likely that concepts have evolved and that new concepts and hybrids of previous methodologies would have been developed. Therefore a more abstract approach is needed.

Giavitto & Godin (2001) describe formalised models in terms of their roles:

- **Pedagogical or heuristic models** are used to illustrate complex relationships between system components and can be used as a teaching tool (e.g. simulation visualisations, causal networks and illustrative diagrams).
- **Normative models** are reference models, which are used as comparisons between scientists or as blueprints for generic illustration of frameworks (e.g. UML models).
- **Constructive models** are used to build upon pre-existing formalisms or approaches. They are sometimes termed composite models and involve a certain level of (sub)model integration.

Such an abstraction does not inform one of any notion of model content. Ideker & Lauffenburger (2003) go further and compare the most popular formalisms in computational cell biology (see Figure 4.1) and classify them by granularity. Epidemiological, i.e. phenomenological, models are placed on one end of the spectrum and mechanistic models on the other.

Definition 4.3 Granularity *(of a model)*
The level of detail that is presented in any model. A fully fine-grained, or low-level, model is defined as the model of which the components cannot become any more fine-

grained, either because the model incorporates a full understanding of microscopic (mechanistic) knowledge, which is of course theoretical, or the model has reached the limit of mechanistic knowledge. Conversely a model is termed course-grained (more phenomenological) when mechanistic knowledge is omitted and behaviour is determined at a higher level(s).

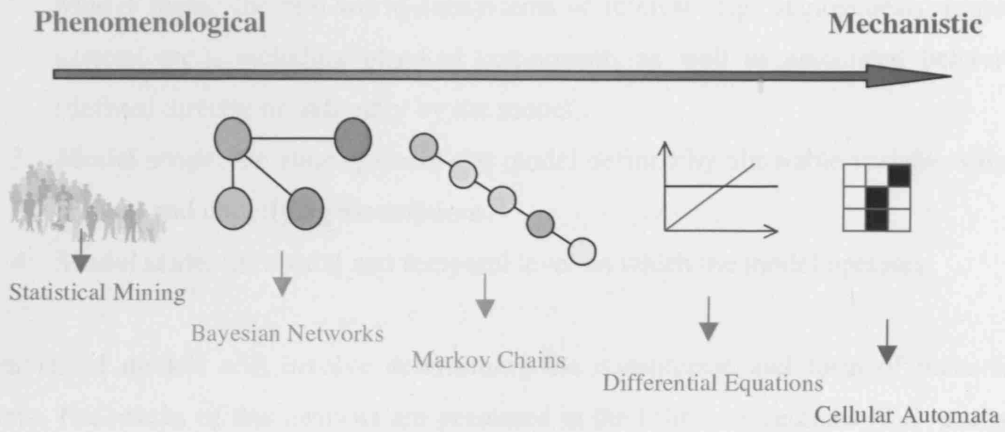


Figure 4.1 Different formalisms pertain to different fields of view. Ideally models and simulations of cancer from which one can ascertain causal and emergent phenomena must come from detailed mechanistic models. Image adapted from Ideker & Lauffenburger (2003).

By itself this crude classification does not shed any light on the model integration problem, except that different models require and contain vastly different volumes of data and data types based on scale (discussed later).

The points made by Giavitto & Godin (2001) and Ideker & Lauffenburger (2003) based on role and granularity-related information content allude to a rough abstraction of model formalisms without actually defining those formalisms (e.g. P/ODE-based, statistical, ABM, etc). There is, however, a need to describe models on a more fundamental basis without restricting oneself to a rigid and formalised classification that cannot be later evolved. Such a description needs to include all of the most salient points of the model in light of the possibility of future integrations with other models: the mathematical, computational and logical formalisms that it uses, how the model is run within a software

environment and associated issues[1], what the model actually represents and how it is to be used (i.e. its goals and limitations). The following criteria has been developed to examine the models:

1. **Model paradigm**: the abstract mathematical and/or computational basis (formalism) of the model.
2. **Model focus**: the real-world subsystems of interest (e.g. angiogenesis, immune system, etc.), including physical components as well as associated behaviour (defined directly or indirectly by the model).
3. **Model scope**: the state space of the model defined by allowable variable values, context and underlying assumptions.
4. **Model scale**: the spatial and temporal level on which the model operates.

Analysis of models will involve determining the constitution and form of these four points. The results of this analysis are presented in the following sections (full results in Appendix A1). These points that define the nature of models, are then further defined from the perspective of model integration and refined into the core challenges that any formalised framework should address. Plausible solutions to these challenges are given in later sections and tested in Chapter 7.

4.3.1.1 Model Paradigms

Definition 4.4: Model Paradigm** and **Paradigm Profile

The fundamental mathematical, computational and/or logical principle(s) upon which a model is based. Models will exhibit more than one paradigm and hence a paradigm profile can be defined. The paradigm(s) dictates not only the nature and behaviour of the model but also the form of any solver that can be attached to it, i.e. how the model is manipulated for use. Note that solvers and models are from hereon treated separately – paradigm, focus, scope and scale relate only to the model in this discussion, though the same theory can be extended to solvers.

[1] This is in fact a matter of solver technology, which will be discussed in more detail later in the chapter.

Definition 4.5: Solver

The general definition of a solver in the literature is a procedure, or set of instructions, that a computer can execute to solve an instantiated model (Tsai, 2001). However this does not go far enough to include abstract constructs, e.g. ODE models that contain uninstantiated variables but can be solved by a series of mathematical techniques that would include curve-fitting and discretisation methods. Therefore the definition is extended here to any enabling mathematical technique or software application that manipulates a model, whether it is instantiated or not, such that the resulting representation is usable in at least partial fulfilment of its potential. Note that not all models are designed to be solvable.

Previous work specifically on paradigms is scant; it seems that a paradigmatic angle of understanding has not been developed though many authors have merely commented on the fact that integrating models across qualitative/quantitative, stochastic/deterministic and discrete/continuous paradigms (sometimes called "dimensions") is a difficult undertaking (Uhrmacher *et al.*, 2005). Villa (2001) has been the closest to addressing this issue albeit in the environmental modelling field with respect to model integration. The approach taken, implemented in the so-called Integrating Modelling Architecture (IMA), makes integration of submodels explicit with respect to paradigm (paradigm in IMA being defined quite loosely as mathematical or computational formalisms, e.g. ABM and ODE). In close correspondence with this view CellML, a formal and accepted standard in systems biology that addresses model exchange standards, models are viewed as compositions of subcomponents (synonymous with submodels) each of which must be described by some metadata[2] that details three fundamental "paradigms": time, space and behaviour (Lloyd *et al.*, 2004). Whilst the explication of time and space is quite straightforward, *behaviour* is more difficult to describe. Villa takes an iterative approach in which a new integration strategy is developed as and when a new paradigm profile is encountered. A semi-autonomous approach to model integration can then be sought in which the IMA software generates translation functions between variables in each model

[2] Grammars are expressed in XML.

component dynamically (i.e. as they are connected together) based on that metadata. Essentially this type of model integration can be termed linear integration, which will be discussed in more detail in §5.3.1. The advantage of such a decomposition is that behaviour encapsulated in individual models need not be lost in the integration process as subcomponents remain as computing subunits within the integrated model. Additionally all variables and models are treated as data sources, which makes the process easier to understand for non-technical users. However it is not entirely clear how complex semantics relating focus, assumptions, etc., can be incorporated into the integration process and, of course, the existence of a translation function is not always guaranteed.

To redefine and interpret model paradigm more formally and clearly an in-depth analysis of the cancer systems biology field has been carried out. The results are as follows (for specific references and examples the reader is referred to Appendix A1).

Functionality is the most abstract paradigm. *Functional* models are referenced within the context of executable mathematical or computational constructs, i.e. models that perform some kind of processing function, but can also refer to more qualitative constructs such as executable semantic networks. Examples of functional models include differential equation models, Bayesian networks, CA and ABM. Though an UML diagram does not perform a specific executable task it is still considered a type of model since it characterises an abstraction of a real-world system, e.g. the realisation of a software system. The same can be said of schemas, which precisely and formally define, for example, the structure of XML documents. These models are herein defined as *non-functional.* Note that an unsolved model, i.e. a theoretical construct, can also be viewed as being non-functional with the given definition. However the concept of solvability is an essential part of the goal of such models, e.g. ODE equations in contrast to schema. One could counter that a UML diagram similarly has solvability at the core of its goal. Essentially a clear dividing line can only be established if one accepts all models as non-functional, whether solvability is a part of the goal or not, and solved forms of models as functional. This entirely supports Definition 2.1.

One can distinguish models further on the basis of *numeric content*. In other words the model is either *quantitative* or *qualitative*. Generally qualitative models that have little or no concept of solvability exhibit the aspect non-functionality. However, with a surge of interest in fuzzy systems wherein qualitative models can be quantified the boundaries of this particular paradigm are becoming increasingly blurred. Fuzzification and defuzzification processes, now commonplace in control system models, have closed the gap between qualitative and quantitative aspects of modelling (Wolkenhauer, 2001). However it must be said that such processes are dependant on human and/or optimised input, e.g. generation of membership functions.

Models also have a *continuity* aspect, i.e. the model is either *discrete* or *continuous*, which more specifically pertains to the actual variables used in the model. In specific modelling paradigm cases this aspect can also pertain to actual components used in the model, e.g. agents within ABS are considered discrete entities (which could concurrently operate on continuous state variables). The previous chapter has highlighted the use of these paradigms in the context of systems biology. Moreover it has become apparent that these extremes are interchangeable when moving from an unsolved to solved form. For example, Anderson & Chaplain (1998) use a model that is fundamentally continuous but solve it discretely via the Euler method. Discrete CA and finite element/mesh methods are common solver technologies for continuous systems (Wolfram, 1994).

Stochasticity is another prominent paradigm, i.e. models have *stochastic* or *non-stochastic* (deterministic) characteristics. A stochastic model implies both functionality and quantitative numeric content. Stochasticity is not strongly linked with continuity even though the traditional probability scale and associated calculi are continuous.

The *centrality* paradigm outlines a fundamental goal-oriented concept within the model-building process and is best understood when considering the difference between simulations and goal-oriented approaches such as optimisation algorithms. For example there is a clear paradigmatic difference between (i) a simulation that is run over time and shows the drug dose effects on tumour growth; (ii) an optimisation or classification

algorithm, such as a neural network, which takes the existing tumour size and consistency as input and calculates the recommended dose requirements to minimise tumour size. The first algorithm runs over time whereas the second algorithm has no notion of any simulated trajectory of target parameters. The second algorithm can also be likened to data-mining algorithms where no simulation as such is executed – rather, an ulterior model or construct is generated, e.g. gene expression data-mining algorithms that infer plausible genetic regulatory networks. Therefore this paradigm is embodied by *centralised* and *decentralised* character. A model can have a specified centralised goal, e.g. optimised therapy or establishment of a classifier from data-mining, as opposed to a relatively decentralised goal that is characteristic of time-continuum simulations.

Finally, there is the aspect of *dimensionality* to all models, i.e. spatial and temporal dimensions. As the previous chapter has shown, models and simulations manifest in a variety of types in terms of dimensionality, from non-dimensional to four-dimensional models (and higher for centralised models). As with centrality, dimensionality is often a property of the solved form rather than the abstract model. For example, ODE equations explicitly defining the Laplacian ∇ can describe dynamics in all three spatial dimensions but may well be solved and simulated in 1 or 2 dimensions for simplicity. Additionally phenomenological state models are often non-dimensional since they describe some observable state (e.g. tumour radius, cell number).

Note that models are characterised by the six paradigms listed above applied to all submodels, i.e. a submodel elicits a paradigm profile. This divided component-based architecture is common to all complex models encountered in the literature, illustrated in Figure 2.1. Formally a model can be defined explicitly as a novel 6-tuple:

$$M = \{P_i^m, \ldots P_n^m\}$$

$$\mathrm{P}_i^m : \begin{pmatrix} F_i^m = \{f_i^m, \neg f_i^m\}, Q_i^m = \{q_i^m, \neg q_i^m\}, \\ C_i^m = \{c_i^m, \neg c_i^m\}, S_i^m = \{s_i^m, \neg s_i^m\}, \\ E_i^m = \{e_i^m, \neg e_i^m\}, D_i^m = \{d_i^m, \ldots\} \end{pmatrix}$$

Formalisation 4.1

where:

M is a model composed of n submodels, indexed by i, with respective paradigm profiles.

P_i^m is the overall paradigm description indexed over i submodels.

X_i^m is the paradigm where F, Q, C, S, E and D represent functionality, numeric content, continuity, stochasticity, centrality and dimensionality respectively.

x_i^m is the paradigm type for X (i.e. f_i^m is "functional", $\neg f_i^m$ is "non-functional", etc).

(Superscripts are used to distinguish symbols from other formalisations in following sections).

Definition 4.6: Submodel

An unsaturated submodel is any model component that can be considered a model in its own right as per Definition 4.1 and that can clearly define each of the six paradigms without mixture (singleton membership) to form a paradigm profile (i.e. the submodel is unsaturated in all paradigms). If the submodel is found to exhibit multiple membership in any one or more paradigms as listed above the submodel is termed saturated. From hereon, for clarity, the prefix "sub" is dropped in the main text unless the context is unclear.

The ultimate challenge in formulating model integration strategies from this perspective is finding a solution, within a single formal framework, for integrating models that exhibit very different paradigm profiles. For example, how can one integrate stochastic and non-stochastic models? The explication of these paradigms is aimed at capturing the most elementary nature of mathematical and computational models, thereby revealing the difficulty of model integration due to paradigmatic differences. If model integration can be understood to be a type of communicative ability between running models differences in paradigm imply the implementation of a translational tool[3], e.g. implementation of a (de)fuzzification algorithm to quantify qualitative models. For any two models, a and b, where $\forall a \wedge b : X_j^a = X_k^b$, and where j and k index submodels that are to be integrated, model integration with respect to paradigms is trivial (excluding the complexities of

[3] As will be explained in following sections, model integration need not solely take the form of this type of chaining.

focus, scope and scale, which are explained later) since paradigms are identical and no translation as such needs to be implemented.

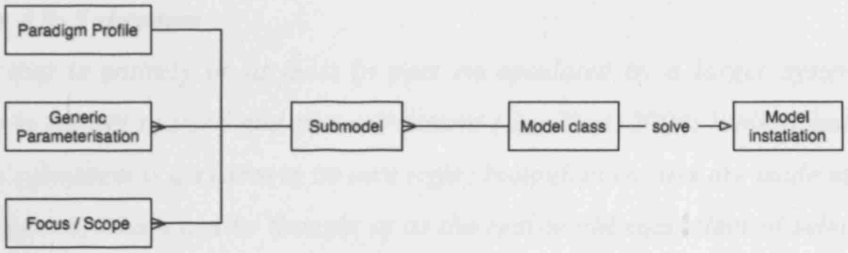


Figure 4.2 a. An entity relationship diagram of model anatomy. As defined in Definition 4.6 a submodel consists of a single paradigm profile. In practice a submodel might not be decomposable as such (e.g. a complex black-box; see Definition 4.12), however conceptually a model can be recursively broken down into functional components that fulfil Definition 4.6. Such submodels almost always ascribe focus and scope and in the preliminary stages (especially so for DE models) are formulated in general forms, which then need to be solved. In the case of DE models, for example, parameters are not instantiated until some solver (curve fitting, integration calculus) can be applied to derive an instantiated version. Hence there is a notion of a model "class" (see §4.4).

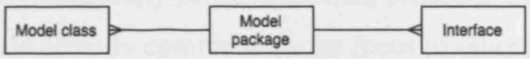


Figure 4.2 b. A model class, or a number of classes, is usually packaged into a functional and accessible component. This could represent a solver container, e.g. an ODE package that can read and interpret an equation and solve given inputs, or a dedicated software package. The interface, i.e. the mechanism of communication and interaction, for this package may not be generic or even fully functional in practice, however it forms a critical part of how the model can be integrated with another. Indeed this interface is key to existing model integration strategies, as will be explained later in the chapter. One can see where the model integration types outlined in Definition 4.2 fit in this view of model anatomy: instantiated, or "live" models, can communicate their state or be interrogated via the interface(s) in what can be termed integrative communication; the uninstantiated form cannot be interrogated or communicated with in the same way – it is not solved – but, as a schematic, can itself be communicated to importing applications via a lingua franca, i.e. isolative communication. Decomposite communication should[4] be more finely-grained in that it takes the submodels of different models and allows them to communicate under a single framework.

4.3.1.2 Model Focus, Scope and Assumptions

Definition 4.7: Model Focus

The artificial partition of a system and its associated observable behaviour such that this partition represents subsystem(s) and behaviour(s) of interest and is accordingly the

[4] Since this integration type is not yet encountered in the literature as per Definition 4.2, it is difficult to summarise this in a general form.

subject of the model. Note that this definition is the abstract equivalent of component (Definition 3.3) with the extension of including associated behaviour.

Definition 4.8: Subsystem

A system that is entirely or at least in part encapsulated by a larger system and its behaviour is subject to itself and its environment (Bar-Yam, 2004; Wolkenhauer, 2001). In effect a subsystem is a system in its own right; biological entities are made up of many such subsystems, which can be thought of as the real-world equivalent of submodel (the entire CS being equated to a complete model).

Chapter 2 identified four overarching modelling foci that are prevalent in the cancer modelling literature, namely growth, response, pathway and angiogenesis models. Whilst it is clear that complex systems can be partitioned into subsystems, system variables and behaviours to reduce the complexity of the modelling procedure, the literature is unclear as to how a model would actually communicate its focus to other models. This is critical from a model integration perspective – an integration must take place in the context of focus, otherwise it is unclear what the integration is actually achieving, i.e. how the integrated model relates to the actual system of interest.

Communication of focal, i.e. subsystem and behavioural, information can be understood to be a metadata problem and as such is coupled to model exchange and communication languages when model integration is considered isolative and integrative as per Definition 4.2. It is even more difficult to describe focus in the decomposite case since its specific solution is highly dependant on focus – indeed, as will be explain in §5.3.3, the precise description of focus is essential in identifying interactors. Essentially this challenge can be distilled down to model semantics that includes information about how a model relates to the real-world – in fact focus amounts to an *in silico* description of what a model represents in the real-world. The interrelation *between* foci yields a definition of how subsystems themselves integrate together in reality (or how they are believed to be integrated) and therefore, when applied to CS, focus correspondingly

becomes complex. Obviously all of these points would play an important role in any formalisation of model integration.

Concisely, the challenges raised are:

1. *How focus is communicated to a model integration environment or modeller*: if any communication mechanism, such as an exchange language, is to communicate model semantics between modelling and/or simulation environments then a formalisation of focus is required to support model longevity, reuse and extension.
2. *How focus is interpreted and processed by the integration framework:* Traditionally a model focus is implicitly given within the model equations or rules (depending on formalism) and/or explained in plain text. Model integration environments and modellers need to interpret focus so that integration can take place within the context of the real system, i.e. integration of models is reflective of integration of subsystems in the real system. Therefore, to ensure the validity of any integration, the integration environment would require a computable form of focus.

Focus is one of the first considerations when formulating a model. Secondarily the limit within which the model is built to operate is hereon referred to as *model scope*. This reduces to boundary conditions on the state space and therefore also defines all allowable values for variables. This incorporates model assumptions, explained below as limitations on how the model relates to reality. Within assumptions one can identify model context, which delimits model use given defined situations. The semantics of model scope can be very complex, even for simple models, and hence it is very difficult to formalise it, which is probably why scope-related concepts are generally written in natural language and rarely tackled formally. Therefore a formalisation has been devised here and is presented in §5.5.3.

Definition 4.9 Model Scope

The operational limits of the model imposed by assumptions and context.

Definition 4.10 Model Assumption

The role of an assumption can abstractly be interpreted as a proposition that is made with the expectation that it will be discharged later[5]*. Given in a modelling perspective an assumption, (set of)* ζ*, can then be defined as a proposition(s) relating to paradigm (e.g. assuming a system is discrete), focus (i.e. what the model abstracts as well as the actual system itself) and context (enforced restrictions) that is taken to be true pending such a time when the model can be revised so as to discharge the said proposition(s). Note that not all assumptions are dischargeable – most notably the ones that are related to model formalism, i.e. paradigm.*

Definition 4.11 Model Context

The operational limits of the model imposed by the real-world information with which the model is parameterised (Finkelstein et al., 2004). This includes the limits placed by model validations.

Scope, in conjunction with focus, covers what modellers frequently term the application of Occam's Razor, i.e. simplification of the system to make the problem more tractable, but this can be abstracted further as follows. Let ϑ represent subsystems, sub/system components and sub/system states:

- **Assumptions on focus (ex/inclusion of ϑ of interest): ζ_ϑ**
    - *Explicit inclusion of focus:* ϑ whose behaviour is included in the model explicitly in the form of variables, constants and expressions.
    - *Explicit exclusion of focus:* ϑ whose behaviour is explicitly excluded from the model; usually this is written in plain text or worked into a solver.
    - *Implicit inclusion (encapsulation) of focus:* ϑ that are represented in the model are implicitly included to encompass finer-grained behaviours and

[5] http://en.wikipedia.org/wiki/Assumption

ϑ. This is equivalent to simplifications of ϑ, e.g. phenomenological models that implicitly include finer-grained dynamics.

- **Assumptions due to paradigm: ζ_P**
  - *Functionality, Numeric Content and Continuity:* These three paradigms often embody simplifications that can be reduced to the other assumption abstractions that are listed here. Functional simulations (the solved form of non-functional construct) stack additional assumptions on the original model by extending or limiting scope. Non-functional models often explicitly exclude focus whilst numerically heavy models tend to be more inclusive. Continuity impacts assumptions on scope (see below) – discrete models are in fact very often chosen for that precise reason, i.e. reduction of state space (Zeigler, 1993).
  - *Stochasticity:* ϑ behaviour is condensed to obey probabilistic laws. This, again, is a simplification of ϑ and is especially made when finer-grained ϑ cannot be simulated due to intractability (e.g. molecular interactions in a large system) or simply because the finer-grained dynamics are not known or cannot be explained by available data.
  - *Centrality:* Global behaviour of ϑ is explicitly formalised and mapped to defined outputs. This can often be viewed as a type of black-boxing, which is a parallel situation to that of simplification into a phenomenological model.
  - *Dimensionality:* Spatial restrictions in relation to focus, i.e. spatial structures, are defined and are often simplistic abstractions of structures in the real system. Such structures, including the concept of time, can also be interpreted as being properties of the solver in some cases.
- **Model Scope (of which model assumptions are a subset): ζ_σ**
  - *Boundary Conditions:* Definition of operational limitations of ϑ.
  - *Model Context:* Atomically this can reduce to the definition of behaviour of a particular ϑ (which can be approximated to a submodel) in specific conditions.

- *Variables and constants:* Empirically deduced or estimated (and therefore with associated belief) variables and constants are assumed into the model and dictate behaviour in the same way submodels dictate behaviour of the parental.
- *Initial Conditions:* The initial values with which the model has been run and subsequently validated. Again, this could be solver-dependant.

Definition 4.12: Black/Grey/White-box

In the case of modelling, simulations and software systems a black-box can be defined as a functional component wherein calculations and/or modifications are made due to an input, which produces a corresponding output; aside from this, no information whatsoever regarding the functional component is exposed. Similarly, a grey- and white-box allow exposure of a certain degree or all information respectively.

Note that all scope-related concepts explained above have been compiled from the tumour modelling literature listed in Appendix A1.

The challenges model scope raises with respect to model integration are:

1. How model scope is communicated, i.e. how is scope metadata encapsulated within a model and how is that metadata extracted and communicated to other models in a standardised fashion?
2. How model scope, specifically model assumptions, is processed.
3. How an integration platform reconciles differences in model scope.
4. When two models of different scope are to be integrated, how differing or contradictory information is reconciled.

Definition 4.13: Scope Processing *(in model integration)*
The computational administration by which an integration platform can expose the plausibility and rationality of specific model integrations on the basis of comparative scope.

The implication here is that a model integration platform must provide mechanisms to interpret model scope and process them such that the integrated model does not violate either of these concepts. Concurrently, as with model focus, it is necessary that the appropriate communication mechanism can express these constructs in a formal manner.

4.3.1.3 Model Scale

As illustrated in Figure 4.3, biological systems operate on a massive scale difference from molecular through to whole-individual dynamics, both temporally and spatially. Modelling at different granularities gives rise to the concept of so-called phenomenological and mechanistic models as described earlier. Models therefore inherently assume a range on the model focus, where range can be defined as a collection of predefined connected foci that span temporal and spatial scales. Herein lies one of the most difficult challenges for model integration strategies. The ultimate challenge is one of tractability – how does a model integration platform integrate models of vastly different scale without incurring a virtually impossible computational cost?

In terms of simulations the multiscale challenge is a universal one that can be found in fields as diverse as the materials sciences (Hyman, 2005), environmental sciences (Roxburgh & Davies, 2006) and geophysical sciences (Iskandarani *et al.*, 2002). Of course the simplest solution is a brute-force approach where the workload of small-scale models is disseminated in a distributed computing environment. However, as illustrated in Chapter 2 by complex small-scale models, even this approach has its limitations when the chosen mesoscopic scale is comparatively large. So-called model abstraction methods (not to be confused with the abstraction of models described earlier) are the most direct solutions to the multiscale problem where the simulations are manipulated to procure faster output without invalidating the model. The objective is to reduce the complexity of

the model by re-conceptualisation (Frantz, 1995). Often this means that sets of functional submodules are amalgamated into a simple black-box that approximates the original set's behaviour. Alternatively whole functional modules may well be deleted from the system altogether to reduce load. Of course increasing efficiency by this process comes at the expense of accuracy[6] and mechanistic insight.

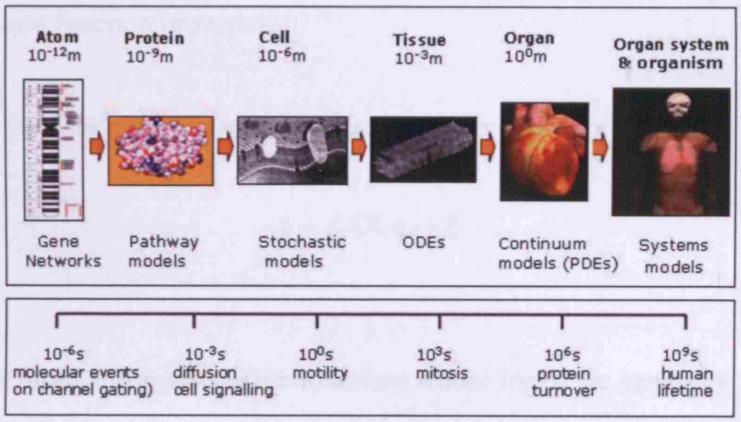


Figure 4.3 The human system can be defined a low level with molecular dynamics that translate as gene and protein interactions through subcellular and cellular pathways. Cells can be interpreted as being on a mesoscopic scale. The entire difference is in the order of 10^{15} and 10^{9} on temporal and spatial scales respectively. Image adapted from Healthcare Review Online[7] and Hunter & Borg (2003).

Model abstraction has evolved into a relatively recent technique called metamodelling[8] (Santos & Porta Nova, 1999) where, essentially, the same principles hold but more formalisation is brought to building models of models (hence the name). Generally a set of equations is sought that transform the inputs to outputs in an efficient calculation. With highly nonlinear systems this is often a difficult task to accomplish, especially when considered in the context of biological complexity where the number of interacting components is enormous and a small number of components, e.g. a small number of gene-regulatory protein molecules, can have a substantial effect on behaviour.

Formally a simulation program or model can be simply expressed as:

---

[6] Santos (1999) and Meckesheimer *et al.* (2002) discuss the issue of precision and validity in detail.
[7] http://hcro.enigma.co.nz/website/print_issue.cfm?issueid=58
[8] This term is also used for other concepts in systems biology. In this thesis the term metamodelling will only apply to the multiscale solution version of the term unless otherwise stated.

$$\mathbf{z} = f(\mathbf{x},\mathbf{r})$$

where:

x and **z** represent vectors of inputs and outputs respectively.

r is a data (or random number, for stochastic models) stream to drive the simulation.

f is the simulation function or model.

Given the above, a metamodel can be expressed as:

$$\mathbf{z} = f_S(\mathbf{X},\mathbf{q}) + \xi$$

where:

f_s is a vector of simpler transformation functions whose forms are generally unknown.

X is the vector of independent variables, e.g. time, and explanatory variables as defined in **x**.

q is a vector of new and/or stochastic variables, treated as abstract explanatory variables, that is yet to be determined.

ξ is an error function imposed by simplification and should be equivalent to $f_s - f$.

Since there are so many unknowns various forms of evolutionary algorithms and statistical methods have been prevalent in this area of research. The form of f_s is often chosen to be some spline, surface response methodology, radial basis function, Cartesian-like correlation model or lookup table (Barton, 1998). Metamodelling at this stage becomes a case of optimisation and curve-fitting. Essentially, if successful, a metamodel simply presents as a phenomenological model.

With the introduction of variables that do not relate directly to the system, **q**, it has been shown that metamodelling can reduce model complexity and increase simulation efficiency with a satisfactorily small error and has already been tried and tested in fields such as the environmental sciences (Khu *et al.* 2004). However it must be said that none

of these applications have tackled natural complexity in any convincing manner – in fact, to the best of the author's knowledge, model abstraction has never been successfully applied to any biological system in any serious way. A strong reason for this simply comes down to accuracy and efficacy. Indeed, Hetherington *et al.* (2006) confirm that simplifications in this manner with respect to liver systems modelling can potentially lead to quite erroneous output and therefore verification techniques such as sensitivity analysis is key in the validation process.

Another interesting concept that shows promise in the multiscale simulation problem is periodic simulation averaging. Patch Dynamics (PD) was first developed by Kevrekidis *et al.* (2003) and bridges the scale gap by averaging the effects of pulsed simulation runs and interpolating between pulses (or patches). Figure 4.4 summarises the formalism. Note that PD is another form metamodelling – a less intensive model has replaced a more complex one by way of interpolation.

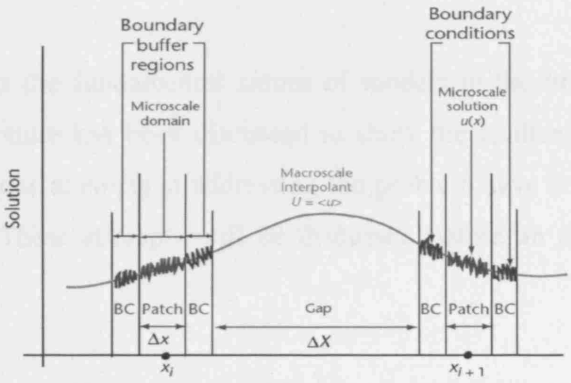


Figure 4.4 Let *u(x)* be the microscale function or simulation and the *U(x)* be the unknown macroscale function. Patch Dynamics is an algorithm that defines small patches where the microscale simulation can be run and averaged. The *u(x)* function, though highly oscillatory on a fine time and space scale (only one dimension shown here) averages to a dynamic that approximates *U(x)*, which is determined by interpolation. Boundary conditions are fitted as "communicators" between which interpolation algorithms infer dynamics. Image taken from Hyman (2005).

Cross-scale modelling has been a subject of research for some time in systems biology, although it must be said that all too often integration of molecular data, e.g. microarray experimental output, with macroscale models have been labelled as "multiscale models".

Whilst such modelling efforts no doubt have their place, in the opinion of the author such labelling is a misnomer.

The momentum with respect to tumour modelling has been relatively gradual. Zhang *et al.* (2007) investigate the macroscale effects of a molecular-based cellular decision mechanism, which gives rise to a migration/proliferation phenotype, with an ABM formalism. With simple expressions for a whole range of stimulatory and inhibitory factors, e.g. TGF, nutrients levels (including diffusion terms) and space constraints, which are parameterised from experimental data the group achieve a good resolution of behavioural dynamics. Fundamentally it is a bottom-up simulation that bridges molecular/cell scales with application of state dynamics to lattice sites, i.e. instead of modelling individual molecular dynamics the behaviour is modelled with simple state-changing rules that simulate molecules *en masse*.

4.4 Model Integration: Previous Work

In the above sections the fundamental nature of models in the tumour modelling and systems biology literature has been discussed to show the challenge model integration really presents. Previous attempts at addressing the problem have been met with varying degrees of success. These attempts will be discussed before an alternative and novel solution is presented.

4.4.1 Model Integration in Management and Environmental Sciences

Model Management Systems (MMS) and Integrated Modelling Environments (IME) were seen as extensions to database management technology (DBMS) [9], and perceived as modelling parallels to the data integration problem (Dolk, 2000). MMS applications were required to enable the modeller to build up models from smaller components in the same way DBMS performed data joins.

[9] For a good review of the parallels between MMS and DBMS the reader is referred to Tsai (2001).

Definition 4.14 Model Integration

(Corollary to Definition 4.2) A protocol that outlines how mathematical or computational components, being functional systems in their own right, are structured coherently together by a formal criterion to build a more representative model of the focus system.

Some of the most well developed fields in this regard are operations research & management sciences (OR/MS) and decision support systems (DSS). In his seminal paper on integrated systems modelling and a further two publications a year later Arthur Geoffrion outlined, to the best of the author's knowledge, the first formal theory of model integration, called Structured Modelling (SM) (Geoffrion, 1987, 1989a, 1989b). Although targeted at OR/MS in particular, and more focused on schema-like integration, the theory of models and graph visualisation of components are to some extent transferable. Uptake had initially been slow but recently SM has received at least some renewed attention from outside the OR/MS community, e.g. the environmental sciences (Makowski, 2005).

The formal SM theory consists of 28 definitions, pertaining to formalised components of models and components that emerge from integrations, and 8 propositions, pertaining to properties and rules of the said definitions, to establish a semantic depiction of a model on three descriptive levels (elemental, generic and modular), which is summarised in Figure 4.5. SM is based on graph theory and therefore captures mathematical modelling by means of components (nodes) interacting (arcs) with each other. Nodes typically either represent entry points for other graphs or a possible link to an alternate model and graphs are further associated with a calling sequence. Model integration thus becomes a task of identification of synonymous nodes between graphs, or possible intermediate-node conversions, and determination of subgraphs that subsequently become obsolete.

Geoffrion identifies four types of model integration, corresponding to four classifications of model entities:

- Model Consolidation or Aggregation (integration of model instances)
- Model Class integration

- Model Paradigm Integration
- Model Tradition Integration

A model class is synonymous with the abstraction made earlier in §4.3.1 wherein a model can exist in an unsolved form. Correspondingly the solved form is defined as a model instantiation, where the constants and variables (except for parametric and dependant variables) are specifically instantiated. The concepts of class and instantiation here have been imported from Object Oriented Programming (OOP). Model paradigms in SM are defined as being a collection of similar model classes, e.g. exponential growth models, and model traditions are defined to be a collection of model paradigms that tend to be associated with one another in a particular field of research. As would be expected, the integration process becomes progressively more difficult as one proceeds through these layers of abstraction listed above.

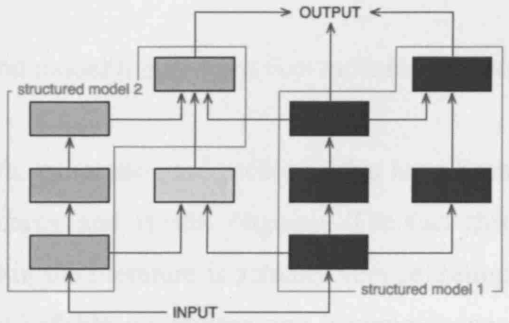


Figure 4.5 In the SM paradigm structured models can be decomposed into elemental structures. This includes O/P/SDE models. There are 5 basic elements: primitive, compound, attribute, function and test (see Geoffrion, 1989, for full explanations). Elements can be connected by relation via the calling sequence, which is the sequence of calculations given inputs, and may well require custom-made "glues" (grey box) or translational elements for each integration. This yields an acyclic graph model. Critically, these individual elements in the IME must correspond to some ontology. An intelligent IME can then begin to integrate given models by relation. Once these elements have been defined, it is possible to view the model in three conceptual levels of abstraction: elemental, generic and modular (progressively through these layers of abstraction the model graph becomes simpler as nodes are collectively grouped).

Note that the input/output connectivity of the graph conforms to the original abstract definition of model integration since all four integration types listed above can clearly be classified as either isolative or communicative integrations. In fact SM represents a linear integration strategy, which is the most prominent form of model integration found in the

literature (explained in more detail in §5.3.1). Though the definitions of model integration as listed above are useful as a rudimentary classification a cross-paradigm, focus and scale integration is not forthcoming from the SM strategy. Indeed all subsequently published papers on SM use, most notably restricted to the fields of management and environmental sciences (Dolk, 2000, Purwasih & Nakamori, 2004, Makowski, 2005, Chari & Sen 1998, Ma, 1998), fail to show SM as an integration framework that can tackle anything more complex than linear systems. Although SM can produce *complicated* mathematical models a formal theory of model integration with respect to *complex*, i.e. non-linear, systems remains as yet underdeveloped (Krishnan & Chari, 2000).

Dolk (1993) identifies further integration types:

- Schema
- Solver
- Environments and modelling systems (i.e. model/simulation software)

Schema, and indeed data, integration are problems that have been extensively researched by the database community and is still ongoing. The fact that solver integration has received some attention in the literature is actually very revealing – most of this research was conducted when affordable computing and accompanying software was only just emerging and procedural thinking was prevalent over modular OOP-like design. Therefore software packages that performed specific model executions came integrated with custom-made solvers that were quite inseparable from the original application. Good software design in OOP nowadays allows one to logically separate components, i.e. solver from model, so that a more appropriate solver can be used for the integrated model.

IME systems take solver integration further, treating them as modelling or simulation engines that can be used interchangeably within a single or multiple models, but not necessarily at the same time (Fricks *et al.*, 1997). Abstracting away from these practical

issues, solvers can be further defined as algorithms that address the solution of a particular model class (defined as in SM). The third point above, integration of modelling systems, will be discussed shortly in terms of software integration. Again, note that all these integration types, as with SM, can be classified neatly into the earlier definitions of isolative or integrative communication.

Mathematical modelling and simulations have a played a crucial role in the environmental and ecological sciences especially due to strong public, political and economic interests in climate change and as such notable attempts have been made at model integration. Argent (2004) reviews the impact of MMS in the environmental sciences in great detail and outlines a number of desirable features that are conducive to model integration, which amongst others include a framework that supports component-based model building, automated service discovery, a well-developed, intuitive GUI and a rich execution (i.e. solver) library. Additionally Rizzoli *et al.* (1998) stress the importance of metadata that should be encapsulated with model subcomponents. This is an important argument as it addresses the fundamental problem of semantic compatibility of models as well as the technical aspect (such metadata could, for example, include information about model focus and scope). As a result of this research a number of IME systems have been developed specifically for the environmental sciences, one of the most recent being COINS (Roxburgh & Davies, 2006). The COINS methodology is quite invasive – a model or simulation is considered to be a package of code and with an appropriate translation layer between the integration platform and all other models and simulations COINS attempts to logically separate mathematical equations from the running code to produce a more compact and efficient single package. Of course such a platform requires an appreciable amount of human guidance since complex packages may well be impossible to integrate by this methodology.

Similarly Villa & Constanza (2000) propose a model integration solution based on a server/client architecture where control of simulations and independent models are distributed over a number of machines and coordination is handled by a common simulation interface via a shared language (communicative integration in Definition 4.2).

Because a common interface and language is used the integration solution is screened from model implementation and formalism (i.e. how the model is being run or solved in its original programming language and therefore this can be considered a kind of cross-paradigm integration). The core of the model integration problem then becomes one of generating bindings for every desired integration.

Environmental science's model integration strategies therefore essentially reduce to a software integration problem (Delden, *in press*; Villa, 2001). In this respect a number of solutions to model integration exist that vary in depth and automation. Multi-Simulation Interface (MSI) is an open-source middleware for simulation communication (i.e. integration by synchronisation) and even message passing interface (MPI) standards can be regarded as an integrating principle in this respect. Taking examples from systems biology, SBW is a service-oriented architecture (SOA) in which multiple applications can communicate and coordinate service requests. CellML and SBML are established model exchange languages in systems biology, something that OR/MS lacks, and provide a pipeline for communicative integration between live software components.

4.4.2 Model Integration in Systems Biology

The Physiome and Integrative Biology projects, described in the previous chapter, and various other *in silico* biology efforts, have yielded some encouraging results with respect to model integration technologies. One of the challenges stated earlier was that of model representation, which still poses a hindrance in fields such as OR/MS. Fortunately the systems biology community has been eager to accept formalisations, as seen with the success of SBML and CellML[10]. Although SBML can theoretically be used as a generalised exchange language it has been primarily used as a representational language for pathways along with other formalisms such as BioPAX (Stromback & Lambix, 2005), whereas use of CellML has been more macro-systems oriented. CellML addresses some

[10] The CellML repository, http://www.cellml.org/examples/repository/, contains over 200 models and a number of SBML repositories, including EBI's BioModel, have been developed.

of the communication problems listed in §4.3.1.2 – model semantics in the form of RDF[11] and mathematical components in the form of MathML[12] can be incorporated within it to describe mathematical relationships between variables (Lloyd *et al.* 2004). However the mathematical capability is thus restricted to the MathML standard, which does not clearly delineate a definitional representation for many methodologies such as ABMs, CA, neural networks, etc., though the creators are acutely aware of that deficiency and are working towards addressing it (Poul Nielsen, University of Auckland, personal correspondence). It is however well suited to representation of differential equation-based models. The hierarchical structure of CellML is modular so as to specifically expose components that are essentially exchangeable or connectable and is thus very similar to the graph-based SM methodology.

Finkelstein *et al.* (2004) take a similar approach in describing models that addresses the challenges described in §4.3.1. The nature of models is taken a step further with a tentative delineation of an ontology of characteristics[13]. This ontology provides the beginnings of a broad formal framework for model description embodied by the development of Composite Model Description Language (CMDL) (Margoninski *et al.*, 2006). Crucially, the framework defines model integration consistency on the basis of constraints placed by assumptions, context, focus and interpretation. This will be discussed in more detail in Chapter 7.

Takahashi *et al.* (2004) provide a more mechanistic approach to model integration, packaged into a software bundle called E-Cell. A meta-algorithm for an integrated model is generated by three components: a *data structure*, which defines informational organisation of a model; a *driver algorithm*, synonymous with SM's calling sequence, defines interactions between submodules; a *numerical integration algorithm* (if appropriate), which defines how state variables are updated (and as the name suggests it is usually a differential equation solver). In the meta-model algorithm mathematical

[11] Resource Description Framework, http://www.w3.org/RDF/
[12] http://www.w3.org/Math/
[13] The term "aspect" in this ontology is synonymous with the term "focus" in this text. Similarly "engine" is synonymous with "solver" in this text. All other terms in the ontology are directly mapped here.

models are defined as vectors of state variables and a set of steppers are defined. Steppers further define processes, which can update variable references via a transition function. Steppers additionally reference local and global times steps, a step function and a vector of interruption methods, which can interrupt step functions of other dependent steppers. A stepper is essentially a computational subunit of a model; therefore an integrated model is ultimately defined as a novel combination of dependant steppers akin to the graph structure of subcomponents represented by CellML and SM. The models are executed in time-steps in a discrete-event simulation. This implementation is elegant but falls short of some important attributes that would define it as a framework for model integration. It is firstly not clear how models of different paradigms, except for discrete/continuous models, can be integrated. Time and spatial scales are dealt with explicitly by a brute-force method (i.e. merely increasing computational power), though some optimisations are possible. Importantly, the semantics imposed by model focus and scope highlighted in §4.3.1.2 and by Finkelstein *et al.* (2004) is not explicitly addressed.

4.5 Summary

The challenge of model integration is clearly defined by highlighting the diversity of models. The origin of this diversity can be found in the differences in formalism, i.e. model paradigm, the processes and behaviours that connect the model to reality (focus) along with associated limitations (scope, context) and complex assumption semantics. Moreover models operate on a wide range of spatial and temporal scales.

The state of the art in terms of model integration is embodied by linear strategies in which the outputs of one model are fed as inputs into another. Structured modelling goes further to classify nodes of the chaining graph so as to encourage model growth by providing, for example, translational functions. In effect current techniques in systems biology attempt to do the very same, however it is apparent that there are at least two views of model integration in this regard – isolative and integrative communications (Definition 4.2) – i.e. models are either communicated by description languages such as

SBML or models are run (solved) with states communicated by a middleware component.

4.6 Conclusion

Model paradigm can be described as a novel 6-tuple. The definitions of focus and explication of scope, context and scale yield an indication of how difficult a model integration exercise can become when these semantics vary from model to model. The implication is that the model integration strategy sought in this project will need to tackle these challenges completely. A decomposite form of model integration is therefore described in the next chapter.

5: A Novel Model Integration Strategy

5.1 Chapter Objectives

- To clearly define the motivation and need for model integration in the cancer systems biology field.
- To re-formalise linear integration (in contrast to previous formalisations in the literature) to give a context to the novel model integration strategy developed in this project.
- To formalise a novel agent-based model integration strategy, which strives to solve the cross-paradigm integration challenge.
- To formalise a novel knowledge-based approach, which strives to solve the cross-focus, scope and context challenges.

5.2 Motivation

Cancer has been dissected from molecular components to clinical phenomenology and so there exists a wealth of data to support the development of understanding of the system at a broad range of levels. Though this has provided the impetus to develop more intricate and complex models in virtually all aspects of the disease there remains a need to produce simulations that provide tangible clinical value, i.e. models that can actually be used as regular components of clinical practice (Hunter & Borg, 2003; Versweyveld, 2006). There are many models that can predict various subsystem behaviours, or individual processes such as signalling networks, but there is a lack of models that can accurately predict tumour behaviour as a whole system. Crucially, whilst in the past biologists have focussed on reductionist views of biological systems, though submodules of the system are inextricably connected, a paradigm shift is occurring such that component interactions on the microscopic scale within the context of the entire system are becoming the focus. This shift provides the impetus for generation of technologies wherein mathematical and computational models of submodule behaviour and properties

are aggregated to function collectively, i.e. model integration. The following points outline why an integrative approach to constructing whole-system models is useful, indeed critical, to cancer systems biology:

1. Whilst a subsystem model can predict the behaviour of its respective focus there is currently no standard formalisation that defines how such models can be integrated aside from linear integration efforts (discussed shortly).
2. A cross-paradigm, cross-scale integrative approach could yield models that combine the strengths of the individual parts, e.g. Stephanou *et al.* (2005) illustrate a custom algorithm that combines the expressive power of an ODE model describing diffusion of TAF and fibronectin uptake with the autonomic advantages of an individual-based model describing EC movement.
3. A cross-focus integrative approach will yield a far more representative model of the system, which is needed to gain a better understanding of the system as a whole, rather than depending solely on a reductive understanding.
4. Whilst models have traditionally been built on the wealth of empirical data available, a more representative cross-focus model will present the opportunity to discover new mechanisms *in silico*, i.e. reverse engineering, as Charusanti *et al.* (2004) illustrate.
5. In the wider scope, validated integrated models could provide the predictive power needed to eliminate, or at the very least assist, the preliminary steps of the research cycle.
    a. From a pharmaceutical perspective this means that preliminary experimentation that cost time and resources can be substituted or assisted, saving on such costs. The overwhelming advantage in this respect is the opportunity to accurately predict, and indeed discover, possible drug targets and dosimetry strategies and could even move therapy closer to individualised medicine.
    b. From a research perspective this affords the opportunity to test theoretical models of mechanisms. Such *in silico* experimentation will aid in advancing the understanding of the system.

c. The tumour system is particularly difficult to observe *in situ*, i.e. in the clinical case, hence much of the data comes from alternate sources, e.g. *in vitro* vascular tissue (Mantzaris *et al.*, 2004, Arakelyan, 2002). It has been shown that mathematical models and simulations can provide a substitute in this kind of situation (Stephanou *et al.*, 2005). An integrative approach aims at establishing models that present such a substitute to close the gap that technological constraints have imposed.

Moreover such an integrative approach is a universal requirement to virtual physiology. These motivations by themselves justify a very in-depth look at the feasibility of model integration technologies.

One can see from the arguments in previous sections that the first two types of model integration outlined in Definition 4.2 are prevalent over the third in all fields of modelling, which is not a surprising finding when considering the fact that integration has been tackled from a cross-paradigm, focus, scope and scale perspective to only a limited degree (this is also very much the case for systems biology). Isolative integration does not require one to be concerned about cross-scale or any other solver/running-related problems since an exchange language defers this duty to the software that will be performing specific integrations; an exchange language does, however, need to be syntactically rich enough to cope with diversity of model types and the complexity of focus and scope. Similarly integrative communication strategies require a translational tool between running models to cope with this complexity and are therefore not particularly concerned with how individual models are solved; however, cross-scale problems become a pertinent issue and the integration platform must provide a specific mechanism to cope with this as well as focus and scope.

Any integration scheme needs to be able to address the challenges and questions stated in §4.3.1. With respect to the three forms of model integration given in Definition 4.2, isolative communication is tantamount to a lingua franca and to a large extent CellML, SBML and CMDL suitably cover this and so this type of integration will not be

addressed directly in this project. Similarly integrative communication takes model integration into a more middleware development setting and has been suitably tackled by software infrastructures such as SBW.

A decomposite model integration strategy strives to take a truly mechanistic stance on model integration wherein all of the challenges described in §4.3.1 must be tackled directly. The following sections detail a novel formalism called Agent-Based Integration (ABI), which embodies such a strategy. Firstly the linear integration strategy, which forms a major component of most if not all current model integration techniques, is reformalised from previous efforts to provide a clear context to ABI. Both the linear and ABI formalisms will be augmented with a novel view of focus and scope that addresses these challenges directly. Associated software engineering, validations and simulations are described in subsequent chapters.

5.3 Novel Model Integration Formalisations

5.3.1 Linear Integration Strategy

Following an analysis of the existing integration technologies, especially those of SM and CellML/E-Cell approaches, it becomes apparent that there is a common integration feature between them. It can be shown that all models are composed of smaller components that are interrelated by a calling sequence. Formally:

$$M_{\phi} : \{m,c\}$$
$$m_k = f_k(\mathbf{x})$$
$$\phi \in \Phi$$

Formalisation 5.1

where:

M_{ϕ} is a model with focus ϕ.

m is a set of n submodels indexed as k. Importantly, submodels also describe subfoci, which is encapsulated by ϕ.

c is a calling sequence on the submodels; when $f(\mathbf{x})$ represents a non-functional component, c becomes redundant.
$\mathbf{x}$ is a vector[1] of v dependant and non-dependant variables and constants in M.
$f_k(\mathbf{x})$ is a function of $\mathbf{x}$ yielding an output vector $\mathbf{z}$. Any f_k need not incorporate $\mathbf{x}$ entirely.
Φ is an (as yet) ambiguous superset defining all foci.

This expression means that model integration as discussed so far reduces to a linear integration solution where inputs from one functional unit are provided by the outputs of another, which can be represented simply as a directed graph, the arcs of which denote the calling sequence. Note that the calling sequence need not be serial. In fact for complex integrations the implementation will almost certainly be parallel, e.g. see Margoninski *et al.*, 2006). It is likely that such complex integrations, especially those that cross large spatial and/or temporal scales, would be run on clusters of computers amongst which whole sections of the calling sequence has been divided – linear integration readily lends itself to this type of parallelisation. This is the mathematical counterpart to software integration, which is the traditional solution to model integration (Geoffrion, 1988) and remains to be so. Indeed looking to SOA's such as SBW, M_ϕ can be compared to services that are chained together – essentially the solution to the model integration problem has distilled to one of communication between services. The applicability of this integration has already proved to be computationally efficient (Dolk, 1993) and black-boxing of components in this way is conceptually very simple.

Linear integration is a partitioning of components in Φ, and subsequent reconnection, such that the integrated model minimises an error function ξ, which has the relation:

$$M_\phi \pm \xi_M \mapsto \psi \qquad \text{Formalisation 5.2}$$

where ψ represents the real system of interest. M therefore represents an abstraction of the real system given a defined certainty or belief as per Definition 4.1. For clarity the error function will be left out in subsequent expressions.

[1] A vector is hereafter treated as an ordered set when appropriate in subsequent explanations.

The goal of model integration is to achieve better *in silico* representations of the real system:

$$\Xi : \{M_\Phi, C\} \qquad \text{Formalisation 5.3}$$

where:

Ξ is a fully integrated model defined by the supersets M_Φ and C.

M_Φ is the superset of all subcomponents connected by C.

C is the superset of all calling sequences on M_Φ.

And ideally:

$$\phi \rightarrow \Phi : \xi_M \rightarrow 0$$
$$\Xi \mapsto \psi \qquad \text{Formalisation 5.4}$$

Ξ represents a model that can perfectly predict the dynamics of the real system. Clearly this is not a practical formulation. Even if the said supersets are known the underlying assumption is that the original models can be decomposed into functional components and the integration of all calling sequences can connect these subcomponents together. Whilst algebraic formulae, e.g. O/PDE, may well be decomposable in this fashion, it is not entirely clear how other formalisms such as CA can be decomposed. Effectively linear integration does not fully address cross-paradigm integration. Moreover the integration is constrained by the number of connectable components. Though functional units can phenomenologically and efficiently represent microscale models, multiscale integration is not specifically addressed.

5.3.2 Agent-Based Integration

The specific requirement in tumour systems modelling is a model integration strategy that does not restrict paradigm use and is able to perform cross-paradigm and cross-scale integrations without violating the semantics of focus and scope. Additionally an ideal

solution will be able to map integrations into a visualisation, such as a 3D simulation visualisation, since analysis from a biologist's perspective is not just numerical but also very much visual – such information can greatly aid the research process. A 3D visualisation provides a real-world perspective of what the model represents and predicts and therefore renders the user with a greater understanding of the results. The Agent-Based Integration (ABI) strategy, along with other mechanisms described here, is designed to address these needs.

Definition 5.1: Computational Common Ground *(for models)*
A mutual shared policy under which models can communicate. Isolative and integrative communication strategies use the communicative language, e.g. CellML or MPI, as a common ground for models.

Thus far model integration strategies have tackled the integration problem by directly, and sometimes intrusively, interacting with various components to achieve some level of assimilation under a formalised framework. In particular, the extent to which linear integration can be performed is largely dependant on the model paradigm, i.e. whether the model can be compartmentalised and subsequently linked. Here it is argued that it is not the model as much as what it represents and its behaviour that is important and therefore integration should take place on this level. Whilst in OR/MS, where the nature of modelling is sequential and to which linear integration is very much suited, the cancer systems biology field presents models that represent processes that run in parallel. In such a situation where paradigms can be very different one must look beyond integration at the level of paradigms. Since it is the behaviour of the models that is ultimately of value, at least in the case of cancer research, a model integration strategy could instead look to using behaviour as the subject of integration. Before one can do this a common ground must be established, i.e. a formalism in which all models can interact regardless of paradigm, focus and scope to which models can be decomposed. Zeigler (1993) in his DEVS application endorses such decompositions when models can be converted through some standardised translation to alternate formalisms that can be interrogated in a discrete event simulation. However such a method presupposes that such a translation is

possible and there is a potentially substantial loss of information when many such translations are chained together. Instead the ABI strategy takes the common ground to the level of an *in silico* metaphor of the system in the form an ABS. Individuals within an ABS can reflectively exist as their real-world counterparts where the common ground is defined as their interactions. To clarify, it is proposed here that individual models can superimpose their behaviour in the form of local interactions onto a single ABS simulation such that the ABS coherently expresses the combined behaviour of the original models. Note that such a superimposition breaks the barrier of paradigms and distils it to a discrete space-time setting in addition to dealing with focus intrinsically (in the form of interactors). This method will be explained in detail below. Also note that linear integration cannot offer such a common ground that is blind to the nature of the model whilst sensitive to its behaviour.

Individual-based systems provide an excellent computational and controllable metaphor for the real system to which other formalisms can be made to adapt to make this grounding possible. Abstractly:

$$\Lambda : \{A, R_A, S\}$$
$$\Xi_\Lambda \mapsto \psi \qquad \text{Formalisation 5.5}$$

where:

Λ is a local interaction-based system (e.g. ABM, CA or a hybrid).

A is the set of all local interactors, e.g. agents or CA cells, and does not have to be static.

R_A is the set of all rules on A, where $\alpha \in A$ and $r \subseteq R$ and r is an ordered set on α, hereon denoted as r_α.

S is the superset of all states (in/finite, discrete/continuous) and state transition functions hold as shown in Table 3.1.

Ξ_Λ is the integrated model as represented by Λ.

Λ can be considered a suitable grounding for another mathematical or computational model M_ϕ under formalism/paradigm λ, assuming Λ simulation behavioural dynamics is

considered over discrete time ($\tau \in t$), if and only if the following is computationally viable:

1. $\forall \tau \in t : M_{\phi,\tau} \xrightarrow{\upsilon} \Lambda_{\tau}$ Formalisation 5.6
   a. Within an acceptable predefined bound of error $\forall \tau$.
   b. υ is a translational mapping function from M_{ϕ} to Λ and is ideally symmetric.
2. $\forall \tau \in t : \lambda_{\tau} \xrightarrow{\Upsilon} \Lambda_{\tau}$ Formalisation 5.7
   a. Υ is a (set of) translational mapping functions to map λ to Λ and is ideally symmetric.

h is defined as the mapping function that enables Formalisation 5.6, and more generally Formalisation 5.7, where:

$$A_{t+\delta} = \left(\forall s_i \in S : s_{i,t} \xrightarrow{r_i} s_{i,t+\delta}\right) \mapsto \Lambda_{t+\delta}$$

$$M_{\phi,t+\delta} = \forall m_k \in M : \mathbf{x}_t \xrightarrow{\partial f_k(\mathbf{x})} \mathbf{x}_{t+\delta}$$

$$(\Lambda_{t+\delta} \wedge M_{\phi,t+\delta}) \mapsto \psi_{t+\delta}$$

$$\partial f_k^{global}(\mathbf{x}) \equiv R_{t+\delta}^{local}(S)$$

$$M_{\phi,t+\delta} \xrightarrow{h} A_{t+\delta}$$ Formalisation 5.8*i-v*

This means that every incremental change in interactors with respect to state changes (including spatial orientation), indexed as i above, is due to the rules associated with the interactors, as per the definition of individual-based systems, and that this change in state(s) contributes collectively to the overall behaviour of Λ (Formalisation 5.6). Moreover, this state change must be reflective of the corresponding state changes given by the original model M_{ϕ} and, more atomically, the submodels that are included in the integration (Formalisation 5.7). Note that essentially this formalism is a discrete solution. Physically this is but a natural implication since all agents within the real system, e.g. biological cells, molecules, etc., are indeed discrete, though discretisation need not be absolute (e.g. molecules can still be measured as volumes or concentrations, which is

stored simply as a state). Though time discretisation is not an absolute necessity it does make the formalisation simpler. Additionally, discrete simulations are known to be more efficient than continuous[2] varieties since, generally, they require less computation (Uhrmacher *et al.*, 2005). The global[3] functions that M_φ defines in discrete time (or continuous time as $\delta \to 0$) map ambiguously via h to local interaction rules R (Formalisation 5.8*v*), which in turn invoke state transitions (Formalisation 5.8*iv*). Indeed if $\Lambda \mapsto \psi$ and $M_\varphi \mapsto \psi$ (Formalisation 5.8*iii*), assuming the formal definition of Λ, then there must be some R^{local} that satisfies the mapping, which is reflective of the original system (Formalisation 5.8*iv*). To create grounding Λ then becomes a matter of finding h. Fundamentally this formalisation is only legitimate if M_φ is indeed considered to be faithful to the real-world system, i.e. it is validated. This assumption reduces the complexity of dealing with contradictory dynamics contributed by different models, though this issue is still resolvable to some extent (explained shortly).

Any formalism, i.e. any mathematical model or simulation that is to be integrated within this framework, must be able to map at least approximately (i.e. within given error bounds) in incremental time and space to Λ whether or not the original model even includes the concepts of time or space. This in turn will inherently depend on the formalism of λ, i.e. the model paradigm – the properties of these formalisations with respect to (sub)model paradigms are summarised below.

Functionality: To the exclusion of those models that are specifically solvable by some solver technology, e.g. an ODE that can be solved by the Euler method, in the case of non-functional models ∂f_k^{global} becomes obsolete since there is no operation to execute that readily maps to Λ and therefore $h = \varnothing$. However non-functional models do contain some information about the target system – for example, even a simple hierarchical model of biological structures in the form a directed graph still directly maps structural

[2] In reality, however, computational representations of numbers are actually always discrete.

[3] Note that though these functions are labelled *global* they need not necessarily be so if λ happens to represent a local interaction system (or other mechanistic system) already, in which case h represents a translation of rules (given spatial and temporal dimensions) from the original system to the current framework.

information about the system. Such information is indeed translatable to Λ, if not to determine any R then at least to identify interactor types. This kind of information is applicable to the knowledge-driven approach described later in §5.3.3.

Numeric Content: Quantitative models obviously hold promise, assuming they are functions, as there is likely to be some transformation function between inputs and outputs by which h can be defined. Qualitative models would be more difficult, especially if the purpose of the model is meant to be purely abstract. Otherwise there may well be precedent for quantifying these models, e.g. application of fuzzy membership functions (Wolkenhauer, 2001).

Continuity: Assuming functionality and numeric content, a continuous model can be considered a special case of discrete mathematics where $\delta \rightarrow 0$. A continuous system should therefore have the capacity to be queried at distinct variable (e.g. time) points. In the case of P/ODEs, for example, one can assume the solution given boundary/initial conditions along a mesh (finite element method). In the decomposite ABI form any intermediate values then would have to be determined by interpolation if they are needed.

Stochasticity: This paradigm presents a greater difficulty than others with respect to ABI. If stochasticity is considered mechanistically, i.e. on a fine scale, the degrees of freedom can be intractably large. Predicting the form of local interactions from inconsistent global behaviour presents another problem. The only way to narrow the number of possible solutions is by constraining parameters through knowledge of the system that does not come from the model directly. This could be achieved by the knowledge-directed approach discussed shortly. When global behaviour is consistent, i.e. in the deterministic case, h can be reliably tested iteratively and its derivation reduces to that of the other paradigms.

Centrality: Assuming functionality and quantitative numeric content a centralised method can be considered to model a defined focus at specific parametric instances, e.g. neural network-based gene expressional diagnostic classifiers or therapy optimisation

methods that define (sub)system states at specific time-points. Often centralised methods as such take instantiated global behaviour and map it to another, sometimes more subtle, global behaviour or end-trajectory and therefore contain very little information pertaining to mechanistic dynamics; one certainly would not begin the integration process with a centralised model, but it may well be possible to direct behaviour so that such a model is satisfied, e.g. attenuation of local rules so that state dependencies become concordant according to a centralised model. The derivation of h therefore requires far more knowledge than what the model readily presents.

Dimensionality: One particular issue that must be addressed is whether any R obtained from a model via h in an x-D setting is at all applicable to a y-D setting, where y represents the dimensionality of the target simulation Λ – the answer to that question obviously depends on what R actually defines. When $x > y$ the derivation of h is conceivable without the need for other information. When $y > x$, however, alternate sources of information may be needed to validate the integrated model. Derivations of h in non-dimensional models reduce to paradigm cases listed above.

The derivation of h is not directly related to focus, scope and scale though there are some considerations that have already been touched upon above. Firstly, the focus of a model will define the specific interactors in Λ, whether they are abstract or actual. This provides the basis on which models can be mapped to, i.e. where R^{local} is applied. In relative terms this is of course associated with scale and how Λ can be run efficiently. For example, to integrate a molecule-level model with a tissue-level model could prove to be extremely difficult since the interactors are defined on vastly different scales. Assumptions and scope impose themselves onto the rules, which will be explained further in §5.3.4 where such model semantics are incorporated into the interactors directly.

Clearly there is no generic or direct mathematical derivation for h, which is heavily dependant on the paradigm. When no direct translation is feasible it seems that an iterative process is necessary wherein local interaction rules are generated and optimised by comparing the behaviours Λ to M_ϕ at each iterative step. Evolutionary techniques lend

themselves to this approach in which an initial set of rules, R_i, which represent educated guesses of the true rule set R, can be enriched. Genetic algorithms (GA), pioneered by Holland (1975), are a proven optimisation technology that has the ability to find global maxima/minima in large search spaces. In fact GA have previously been applied to CA to optimise rules to achieve global behaviour (Kanoh & Wu, 2003). Mitchell *et al.* (1996) investigate the feasibility of evolving two-state 2D automata to achieve specified global behaviour and report some success though the sheer number of possible solutions, which exponentially increased as the size of the CA and number of rules increased, admittedly exceeded even the capacity of their GA implementation. This difficulty is well established in the literature however when the solution space is constrained by restricting rule numbers and forms, neighbourhood radii and state magnitudes global behaviour optimisation is indeed achievable. Colonna *et al.* (1998) demonstrate this by learning CA rules that simulate coordinated social and vehicle movements although any CA learning on the level of biological complexity, to the best of the author's knowledge, has not yet materialised in the literature.

What is obvious from relating h to the paradigms listed above is that there is an explicit need for some knowledge about focus (ϕ), interactors (ι), and states (S) to arrive at a solution as well as to limit on the number of plausible solutions. Therefore an explicit knowledge-driven approach is now proposed, which not only drives a search for a solution to h but simultaneously attempts to manage model focus and scope problems that arise in model integration strategies.

5.3.3 A Knowledge-Driven Approach (KDA) for Addressing Model Focus and Scope

A framework in which focus can be expressed will greatly aid the modeller in both understanding what models represent as well as how they should be integrated. Such a method must be able to describe the relation *between* all foci as well as each and every model focus itself. None of the model integration platforms described earlier tackle focus directly even though it forms a fundamental part of the model – in essence it is the connection the model has with reality and so development of a method that can express Φ in a model integration strategy is highly desirable.

The ideal properties of such a method can be defined:

1. High level of capture of the semantics of models.
    a. As description of focus given in §4.3.1.2.
2. Direct and dynamic mapping of actual system components.
    a. The representation should fully reflect the original modeller's knowledge of the system, i.e. what is termed "expert knowledge" in the literature.
    b. The representation should be robust to change in or introduction of new knowledge.
3. High level of expression with a relatively simple representation to avoid error.
    a. Flexibility in what can actually be represented is vital as complex models of complex systems will presumably go hand-in-hand with complex semantics. Moreover these semantics may well incur human error when the (graphical or exchange language) representation is difficult to understand or generate.
4. Portability.
    a. Preferably the knowledge that Φ represents should be portable to alternate applications, e.g. via some shared standard.

In this proof-of-concept the representation format[4] itself is not as important as its underlying theoretical basis; since focus has thus far been discussed in terms of partitions of the system a set theoretic approach is taken here.

[4] There are a handful of techniques that can manage focus with the said requirements such as semantic networks (SN) (Marinov & Zheliazkova, 2005), sometimes called topic/concept maps in the literature, and ontologies realised as, for example, DAML-OIL (http://www.daml.org/) or OWL (http://www.w3.org/TR/owl-features/) (and associated knowledge-base technologies). Even though the latter are technically designed for semantic web applications, they provide a neat and well-defined, logical representational format and are widely accepted ontological technologies albeit difficult to understand syntactically. On the other hand SN, though far more versatile due to reduced formalisation, are not as portable via any widely accepted format but are relatively easy to understand and build. Purely for simplicity and versatility, as well as representing the modeller's beliefs about the system, SN is the representation of choice but for portability OWL is superior.

Given the current state of the tumour modelling literature Φ is evidently non-existent and cannot be explicitly sought because of a lack of complete knowledge (if indeed one can assume *complete* knowledge is even attainable) – one of the major reasons for this is technological limitations, i.e. many of the fine details of tumours *in vivo* cannot be measured accurately. Moreover a realisation of Φ would ideally require the agreement of a number of experts for a satisfactory level of accuracy as well as acceptance by the field's wider community. To the best of the author's knowledge no such organisation exists although the Integrative Biology Consortium has recently addressed that very issue[5]. Besides this the current situation and requirements suggest that realising Φ is therefore an unfeasible undertaking at present

To make a solution to this challenge practical it is proposed that Φ be expressed iteratively as Φ_I, where $\forall \phi \in \Phi_I \subset \Phi$. Φ_I can be viewed as a theoretical partitioning of the system based on individual expert (i.e. the user, modeller or panel) knowledge given a set of individual models. Simply put, as one integrates a model into another independent model (in the case of the first integration) or an existing integrated model, Φ_I iteratively develops (tending to Φ) and encapsulates all foci encountered in an ongoing integration exercise.

For ease of expression it is noted that all complex models inherently contain submodels and so the "sub" is dropped here and any reference to "model" can be taken to be equally applicable to submodels in all formalisations that follow. Additionally note here that constants and variables can be treated equivalently if and only if the basic requirement of conceptualisation of a defined behaviour is fulfilled since, as will be explained below, focus will be described abstractly from a behavioural viewpoint.

Models relate content to a specific set of real-world behaviours and real-world system components, which completes the abstract definition of focus given in §4.3.1.2. It then naturally follows that Φ_I should be expressed in terms of real-world behaviours and how they relate to real-world objects. The anatomy of ϕ consists of a system behaviour, which

[5] Nagl, S., personal correspondence.

the associated mathematical component conceptualises, and the many entities that collectively yield the entire system (or partition thereof) and contribute to the said behaviour.

Let ϕ represent the focus of a model or variable, which relates a behaviour to a real-world component κ, or a group of real-world entities that can be coherently categorised as an individual (e.g. an organ, though made up of many components, can be categorised as an individual when given in the context of a whole-body simulation). It follows that Φ_I iteratively builds up a hitherto faithful *in silico* metaphor of the system. Furthermore an interesting property of complex bottom-up systems (§3.3) is that behaviours can be further dissected into a number of more fine-grained behaviours, which themselves will be associated with a finer-grained set κ, etc. Concurrently, it has already been demonstrated that this is indeed an inherent assumption of simulations and models that, sometimes implicitly, metamodel finer-grained behaviour into a simple black-box. As a matter of fact this *behavioural inclusion* assumption is part and parcel of the corresponding ϕ if $\phi \mapsto \psi$ holds true. For example, a Gompertz model (M_ϕ) of tumour growth (ϕ) is a phenomenological model of a relatively macroscale behaviour (i.e. tumour growth dynamics, $\phi \mapsto \psi$ holds for MTS) and as such implicitly includes finer-grained behaviours such as pathway dynamics, cellular dynamics, molecular dynamics and so on. This can be intuitively visualised as a directed acyclic graph of behavioural inclusion where nodes of the graph are specific behaviours and directed arcs denote inclusion; a node is a superset if there exists a tail arc that leads to another behaviour, i.e. a subset, and if not is simply defined as a singleton (where the one member is an instantiation, explained shortly). In essence a *Behavioural Inclusion Tree* (BIT) can be developed.

Definition 5.2: Behavioural Inclusion

Consider foci to be viewed in a set-theoretic manner; behavioural inclusion can be defined as identification of behaviours (sets or singletons) that are wholly (crisp) or partially (crisp/fuzzy) subsets of more macroscale behaviours that wholly or partially

represent the behaviour of the subsets. Presently, for clarity, only crisp sets will be considered. Note that a behaviour (set or singleton) can be a member of a number of sets in which case it partially accounts for those behaviours without having to consider it fuzzy.

As seen in Chapter 2 there are two important concepts of focus and associated behaviour that one must define: the first is that of behavioural inclusion as explained above; secondly there exists an abstraction of behaviours, in terms of an ontological perspective, from model to model that must be dealt with directly to wholly capture the semantics. For example, there are various types of dynamics such as molecular flux, EC and other cellular motility, blood flow, and even growth itself, and within these concepts there are subtypes of dynamics that can be defined. The aim is to enable one to distil the most abstract behaviours of a biological system, and idealise even further to complex systems in general, so that the resulting framework need not be recreated for new integration exercises that may involve systems other than those thus far encountered. This facilitates the iterative building of Φ_I as explained earlier. Simultaneously the integration platform needs to define the most specific behaviours as in the BIT. A set-theoretic/OOP[6] approach for model focus affords this power of abstraction so that one can deal with these two concepts concurrently.

Let U^a and U^b be partitions over the universe of discourse that represents the behavioural class hierarchy, i.e. abstractions of behaviour, and the BIT hierarchy respectively. This partition is also made to highlight the difference in semantics of the subset property – in the case of abstraction the subset property is equivalent to inheritance, i.e. the usual *is-a* OOP relationship, whereas in BIT it denotes behavioural inclusion. Behaviours in the abstracted sense can be considered OOP-like classes whereas behaviours that models actually exhibit can be considered OOP-like instantiations, θ_k. In this way both behavioural abstraction and BIT can be represented on the same graph. Formally:

---

[6] Only the concepts of inheritance, classes and instantiations are considered here since these concepts are proven and powerful tools to describe abstraction and are integral concepts in the knowledge representation technologies such as SN (Suh *et al.*, 1993) and OWL.

$$U = U^a \cup U^b$$

$$\phi : \{\theta_0 ... \theta_n\} \text{ where } \forall k \theta_k \in U^a \wedge U^b$$

The complete model focus (ϕ) is defined in this framework as the set of all instantiated behaviours that are both part of the abstraction tree and BIT. Behaviours can be collectively defined as system concepts (most abstract ϕ), which involve system entities (most abstract κ). Systems concepts themselves are non-entities. System concepts subsume (include, as defined above) other system concepts to create a BIT. Additionally, to include model scope system concepts are restricted by system constraints (Definitions 4.9-11). These abstractions are summarised in Figure 5.1 in which the set of all core classes, c, and related properties is shown. This is a novel construct in which behaviour and system components can simultaneously be described, which will be illustrated and validated in the next chapter.

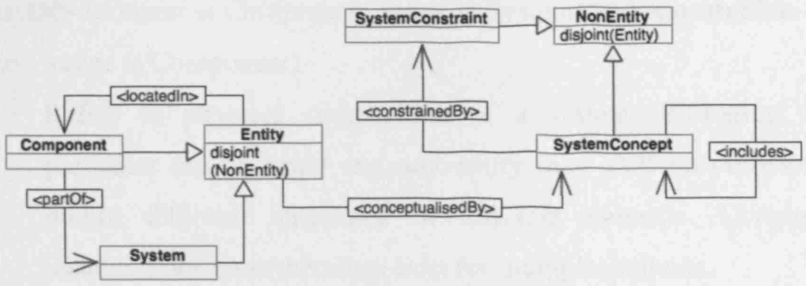


Figure 5.1 Abstraction of the real system and relational mapping of model focus (core classes). Taking an OOP approach, focus itself is conceptualised by a SystemConcept class from which other concepts inherit and/or form a BIT. Note that these relationships also form a type of system ontology. All solid arrows without a label denote inheritance, in terms of abstraction, from parent (head) to child (tail).

Core classes are defined as:

- **SystemConcept**: Any NonEntity concept of a real-world system that describes some behaviour that can be conceptualised by a model, e.g. a biological process.
- **SystemContraint**: Any NonEntity concept of a real-world system that constrains the behaviour of a SystemConcept. This extends the definition U = Φ_I to include scope and context that can be symbolised by any derivation of SystemContraint. Hence a SystemContraint can represent logical or mathematical (but not physical)

constraints. Mathematical constraints in particular are important since these include parameterisation, e.g. curve fitting, which encapsulates assumptions about the model that is otherwise difficult to formalise in a computational construct.

- **Entity**: Any physically observable element of the real-world.
- **NonEntity**: Anything that is not an Entity.
- **Component**: Any Entity that is part of a real-world system.
- **System**: The observable real-world system composed of Components and whose dynamics are defined by behaviours, i.e. SystemConcepts.

Core properties[7] are defined as:

- **<includes>** (both domain and range is SystemConcept)
  - Equivalent to Definition 5.2.
  - Transitive.
- **<partOf>** (domain is Component, range is System) and **<locatedIn>** (domain is Entity, range is Component)
  - Refers to physical composition of a system. Biological systems in particular are dynamic and any entity may well be compartmentalised within different structures in different contexts. Component class instantiations must therefore exist for multiple contexts.
  - Transitive.
- **<conceptualisedBy>** (domain is Entity, range is SystemConcept)
  - Refers to a realisation of process by which a physical entity can be conceptualised by a model, i.e. a SystemConcept.
  - Transitive.
- **<conceptualises>** (domain is SystemConcept, range is Component)
  - Inverse of *<conceptualisedBy>*.
- **<constrainedBy>** (domain is SystemConcept, range is SystemConstraint)
  - Refers to the scope and context of a SystemConcept.
  - Transitive

[7] Note that all properties are enforced on defined domains and ranges.

Inheritance enables desirable properties when used in conjunction with other relations as listed above. For example, since a child class, *child*, inherits all attributes from all parental classes, instantiations of *child* will also inherit all transitive (and other) relations that parental classes possess. Due to the integration of these OOP concepts the said properties that are defined on instantiations are enforced. In other words, an instantiation of any of the said concepts *must* define the properties imposed by the abstract framework (Figure 5.1) with other instantiations of the correct type. Therefore this method, herein referred to as the knowledge-driven approach (KDA), enables description of an *in silico* metaphor of a real system to any level of abstraction within the limits of (inputted) knowledge.

5.3.4 Knowledge-Driven ABI: A Novel Formal Protocol for Model Integration

ABI and KDA have been formalised separately but these two methods can be used together to form a specific solution to model integration. Whilst Λ, together with h, represents a framework that can tackle the cross-paradigm challenge, KDA directly addresses the model focus challenge, including any scope and context that can be symbolised as a constraint, and model assumptions imposed by focus, ζ_ϕ. ζ_σ can also be integrated into Λ in the form of rules; boundary conditions impose themselves as limitations in rules that regulate behaviour and context is similarly treated as explained in §4.3.1.2. Since h incorporates behaviour due to paradigm, ζ_P is also covered by Λ. A fusion of these two approaches is described in which the full range of challenges can be addressed. As described earlier, the key to addressing the multiscale problem is either application of the brute-force approach or some metamodelling scheme such as patch dynamics and is therefore a problem associated more with the enabling software.

ABI establishes $\kappa \equiv \iota$ where $\iota \in \{\alpha_{all} \vee \chi_{all}\}$ and α_{all} and χ_{all} are supersets of all agent/cell individuals (R equally applies to κ). Let R, representing the rules derived via h, be partitioned into rule subsets, ρ_m, that individually represents the rules that are direct derivations due to submodel $m \in M$. Simply speaking, ρ_m defines the behaviour of ι, and therefore κ, as local interaction rules due to m, in the time step $t+\delta$ to achieve a global

dynamic that is consistent with (i) the original model in the same or similar time step and (ii) the real-world component κ, to some predefined degree of accuracy.

The d and μ functions (as given in Table 3.1) make a choice of which rules to fire for any ι given a local and/or global situation[8]. d and μ can therefore be formally extended such that:

$$\begin{aligned} Action : d(R,S,E,G) &= \rho_m^{action} \\ \varsigma_\iota &: \{S,E,G\} \\ \therefore \rho_m^{action} &= d(R,\varsigma_\iota) \end{aligned} \qquad \text{Formalisation 5.9}$$

where:
G is the set of all global states of Λ.
ς_ι is the context of ι (embodied by S, E and G as per the formal definition).

Model scope, m_σ, can be incorporated into this methodology by integrating boundary knowledge of m into ρ, d, S and/or G depending on what boundaries are defined. When applied to d or the rule set of ι, m_σ restricts ρ in such a way that the resulting dynamics in Λ will not violate the boundaries σ represents. Similar is the case for S and G, i.e. state dynamics are restricted so as to not violate σ. In fact σ signifies an additional behaviour for ι, and consequently Λ, that h must codify into local interaction rules. In short, an ABM simulation can be built from a model given enough metadata and knowledge about M_ϕ and ψ respectively. With each progressive integration, the ρ subsets must be ordered such that they are concordant with respect to focus (they represent different behaviours) and scope (application of d), otherwise an integration of the rules is required, which is detailed below.

For simplicity, let the formal integration be applied to submodel m_l where only a single paradigm profile exists and κ and ϕ (as a full KDA specification) are known. In the case

[8] For clarity only d will be considered though the following formalisations are equally applicable to μ.

of the first model that is to be represented in Λ, i.e. Λ is *naive*, h is simply invoked to find $\rho^1_m = R_\iota$. Note that though subsequent integrations yield the sets $\bigcup \rho^i_m = R_\iota$ any $\rho \subseteq \rho^i_m$ can be shared, i.e. an existing rule in the rule set attributed to a previous model integration is allowed to form any part of the solution of a new integration so far as there is no incompatibility issue.

Let each submodel, with corresponding κ and ϕ (as a full KDA specification), that is to be integrated into Λ undergo this decomposition process separately within a defined bound of error. If one model conceptualises any SystemConcept that is already wholly or partially conceptualised by another model, i.e. there is an *overlap* (intersection in the separate KDA specifications of each model), or contributes any ρ to an existing ι in Λ then a synergy must be reached between the models such that there is no contradiction in behaviour of ι. In Λ this means that each ρ contributed by each m contains rules that contribute to the same behaviour and so this overlap in contribution must be reconciled. The four possible combinations are as follows between two submodels, m_1 *and* m_2, hereafter defined as RC1, RC2, RC3 and RC4 (Rule Combination n):

1. **Exact overlap:** m_1 and m_2 cover the same behaviours.
   a. This is a model substitution/replacement situation. Ideally, since both models are abstracting the same subsystem assuming $m_{1,2} \mapsto \psi$, their behaviours should actually be the same and their respective ρ sets should not clash in terms of behaviour of ι. However, if m_1 and m_2 represent theoretically alternative models (e.g. two possible models of a pathway whose dynamics can only be hypothesised into m_1 and m_2) some reconciliations can be made in terms of corresponding ρ:
      i. Reject $\rho_1 \vee \rho_2$ and perform two separate simulations. This strategy would be invoked if it were determined that m_1 and m_2 are irreconcilable because they each represent two very different and distinct beliefs of the subsystem and κ.

ii. Accept $\rho_1 \wedge \rho_2$ in a decision framework. This strategy would be invoked if it were determined that m_1 and m_2 represent very similar beliefs of the subsystem and κ. d must now additionally determine under what circumstances ρ_1 and ρ_2 are fired.

iii. Accept $\rho_1 \wedge \rho_2$ in a belief framework. This strategy would be invoked if it were determined that m_1 and m_2 represent very similar beliefs of the subsystem and κ. Essentially ρ_1 and ρ_2 are now fired under some belief mechanism other than a crisp decision process, i.e. both rules are fired to a certain degree. This can be viewed as a fuzzy firing mechanism.

2. **Partial overlap:** m_1 and m_2 partially cover the same behaviours.
   a. Essentially this is the same as point 1 except that ρ_1 and ρ_2 are now partitioned into a shared subset $\rho_{1\wedge 2} = \rho_1 \cap \rho_2$ where each overlap subset $\rho^1_{1\wedge 2} = \rho_{1\wedge 2} - (\forall r \in \rho_2)$ and $\rho^2_{1\wedge 2} = \rho_{1\wedge 2} - (\forall r \in \rho_1)$ can be treated as the exact overlaps described above so that the same reconciliation strategies listed above can be applied.
3. **Total overlap (subsumption):** m_1 totally subsumes the behaviour that m_2 represents in addition to other behaviours not included in m_2.
   a. This is analogous to the situation described earlier wherein a phenomenological behaviour totally includes, i.e. subsumes, more mechanistic behaviour, however note that the behaviours need not be more/less mechanistic with respect to each other. In the case of an iterative integration (i.e. the process in which m_2 is being integrated with m_1 where m_1 is already represented in Λ), ρ_2 *must* perform in Λ such that it does not interfere with the behaviour elicited by ρ_1. However in the absence of the set $\neg(\rho_1 \wedge \rho_2)$ this is not always practical to achieve. In such a case a rule mergence may well be the best solution wherein the behaviour of the integrated model is as close to m_1 as possible.
4. **Null overlap:** m_1 and m_2 elicit behaviours that are separable.
   a. If two models elicit behaviours that are separable their respective BITs do not integrate, i.e. there are no possible connections. I_1 that has been derived from

m_1 and I_2 that has been derived from m_2, $I_1 \neq I_2$ unless there exists pairs of $\iota \in I_1 \vee I_2$ that interact in Λ.

Note that linear integration is also feasible in this formalism in at least three modes:

1. $\forall m_k \in M_\phi$ are integrated into Λ as described above.
2. An entire M_ϕ, if its behaviour can be encapsulated into any ι in Λ associated with a set of rules (therefore directly able to affect states S as required).
3. A parallel approach can be adopted wherein the integration is minimal: Λ runs separately from M_ϕ, which simply directly affects S and/or G.

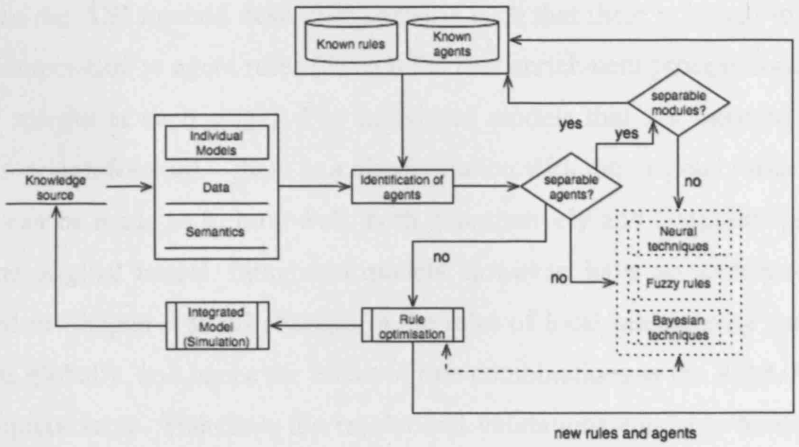


Figure 5.2 Knowledge about the model itself as well as external knowledge contribute to the model integration process by KDA, feeding into the ABI process as described in the main text to identify key interactors and behaviours. Rules are optimised and saved for future integrations or further optimisation. Critically, the process preserves the semantics and behaviour of the inputted models. Neural and Bayesian techniques and fuzzy rules could be used in this process as metamodelling techniques, which will be discussed more fully in Chapter 8.

Unifying all of these concepts yields the protocol summarised in Figure 5.2. Firstly, as with any ABM simulation, a mesoscopic scale on which ι is defined must be identified. For different models this scale may well be slightly different, within reasonable bounds, depending on the anticipated set I. The entire simulation framework Λ is built around this mesoscopic scale wherein more global (i.e. more macro) behaviours are considered emergent of the small-scale subset of I and finer-grained (i.e. more micro) behaviours are either defined explicitly within the framework (brute-force method) or inbuilt

(metamodelled) into the rules and/or states (and therefore state transitions) of the larger-scale subset of *I*. This is analogous to Brenner's so-called middle-out approach, which is an accepted albeit informal approach in system biology (Noble, 2005). Secondly, the *h* function is applied to the models progressively as explained in §5.3.4, which yields the respective rule sets ρ_m. Every model must also be consecutively encoded by KDA to generate the BIT, which defines the behavioural overlap between the models. When this is known the rule integration strategies, as listed above, can be applied to give an ABM that is representative of the original constructs.

5.3.5 Expected Results and Validation

An important question is how far an integrated model can be validated mathematically. The nature of the ABI method described above is such that there is bound to cumulative error in decomposition to agent rules (though the rule enrichment process should ensure a satisfactory margin at each stage). For individual models that are integrated into ABI validation is straightforward – there is a direct relation with the original model and hence an analysis can be made as to how well, both quantitatively and qualitatively, the ABM simulates the original model. Integrated models, however, have no such direct relation. As explained in Chapter 3 small changes in the rules of local interactivity can have non-linear effects globally, and hence the effect of rule combinations in the KDA/ABI method could be unpredictable. Therefore the results and validations can only be qualitative in nature, with comparisons that can be drawn from the original models and known knowledge of the real system.

5.4 Summary

A formal methodology is proposed based on the literature review presented earlier, wherein an analysis revealed the anatomy of models: abstractions can be made with respect to paradigm, focus, scope, context and scale. Note that though these abstractions are made from the tumour modelling literature, they do in fact pertain to basic mathematical principles. To address cross-paradigm integration the ABI strategy is proposed whereas focus, scope and context are addressed by the KDA approach. Cross-

scale integration is left to the implementing software system. Together these formalisations can be distilled into a protocol in which models can be integrated.

5.5 Conclusion

By using generic classes that include modelling concepts and their corresponding real-world counterparts one begin to build a behavioural inclusion tree that describes behavioural subsumption. Model integration can then begin from a behavioural perspective by additionally optimising agent rules, which effectively takes into account the different types of model focus, scope, context and assumptions and is additionally paradigm-free with respect to the original models. The efficacy of this approach will be tested in the next chapter.

6: SOFTWARE DESIGN AND DEVELOPMENT

6.1 Chapter Objectives

- To detail functional requirements for the intended software application.
- To detail software requirements to complete a proof-of-concept of the model integration strategy introduced in the previous chapter.
- To detail the software development done, including rationale.
- To detail software profiling and testing results and thereby verify its efficacy so that further experimentation can be carried out (next chapter).

6.2 Functional Requirements

MMS requirements are aimed at a software system that fully supports the model lifecycle including model integration and simulation, however currently the immediate aim is the production of a proof-of-concept application that primarily demonstrates ABI/KDA. Therefore functional requirements are reduced to:

- Rule enrichment environment.
    - Pluggable and configurable GA environment.
        - One should be able to "swap" components within the infrastructure, e.g. fitness function implementation, and have these components configurable (without the need for recompilation).
        - The GA module should be integrated, but not tightly coupled, to the simulation environment.
            - Achievable by developing these two modules (i.e. optimisation module and simulation environment) as separate applications.
        - Distributable.
- Simulation environment.

- Agent and simulation space flexibility.
  - Static and non-static agents, lattice sites, associated state memories, configurable sensors and actuators.
  - Configurable simulation space including dimensionality, neighbourhood calculation strategies.
- Distributable.
  - Agents and simulation spaces should be able to operate on any number of machines and function concordantly. The nature of the parallelisation should be configurable.
- Rule execution.
  - Simulation implementation-independent rule execution language that is configurable via the optimisation method and rich enough to allow a wide range of executions.
- Logging.
  - All simulation states should be configurable so that trajectories can be persisted and analysed *a posteriori*.

A user-interface for the simulation environment, i.e. a 3D output and control panel, is not considered a priority provided that suitable outputs from which meaningful results can be drawn are produced.

6.3 Software Design and Results

In a wider scope the model integration platform described here is intended to form part of a model management system that supports systems biology-based modelling. However, for the purposes of a proof-of-concept the requirements are restricted as follows.

To create a package that will be extensible to include all of these points will require a design that is:

- Modular and non-duplicate design (Thomas & Hunt, 2002).

- Efficient and robust to change.
- Intuitive for human readers, which additionally requires detailed in-code documentation.

Loosely coupled modularisation can be achieved by development of common interfaces[1], which most modern languages supports, that are designed to fully enable a broad range of functionality and access. With due regard to the requirements listed above the following components[2], each of which will require appropriate programmatic interface, can be identified:

- **Initiation and Distribution Module:** Provides all the classes necessary to begin the model integration process, including start-up services, e.g. initialising participating computers, and initial (pre-runtime) distribution strategies.
- **Server/Client Architecture:** A set of interfaces with default implementations that define direct access to different JVMs.
    - **Communication Module:** Handles communication between running participating computers.
- **Simulation Engine**
    - **Global Synchronisation Components:** Coordination of rule-firing and agents (discrete time across the entire simulation as well as movement).
    - **Space-Time Definitional Components:** Library of interfaces that define space-time attributes of the simulation such as dimensionality, coordinate systems and time steps (ts). The resulting simulation should be able to run ABMs as well as CA under the same framework.
    - **Predefined Avatar Components:** a library of abstracted, extendible interactor classes that occupy the simulation.

---

[1] Programmatic interfaces, not to be confused with graphic interfaces, are special Java classes that only contain method declarations and no implementations of those methods. If any class `implements` an interface then an object of that type can be dynamically cast as such and is required to define all declared methods.

[2] (Sub)components can be defined as entire replaceable functional units that perform a specific non-duplicate job within the system.

- **Computation Distribution Components:** Protocols that define how computations should be distributed at runtime and related optimisation options.

- **Rule Engine**
  - **Rule Parsing Components:** A set of interfaces defining how rules can be parsed from whatever form they exist into antecedents and consequents.
  - **Rule-Firing Components:** A set of interfaces for Rule interpretation and execution including any special handling procedures, e.g. defuzzification.
- **Rule Optimisation Methods**
  - **GA Components:** Engines that define GA mechanisms (mutation, mating, etc.) and related protocols, e.g. distribution of optimised rule sets.

The resulting software is tested on a small Beowulf cluster (see Appendix B1 for specification) and is intended to migrate to the University College London Keter cluster[3]. Both of these clusters provide good support for high volume network traffic (Keter more so at up to $5Gbs^{-1}$), which tallies well with the basic simulation architecture described below.

The basic design of the required application is illustrated in Figure 6.1. Details are given in following sections.

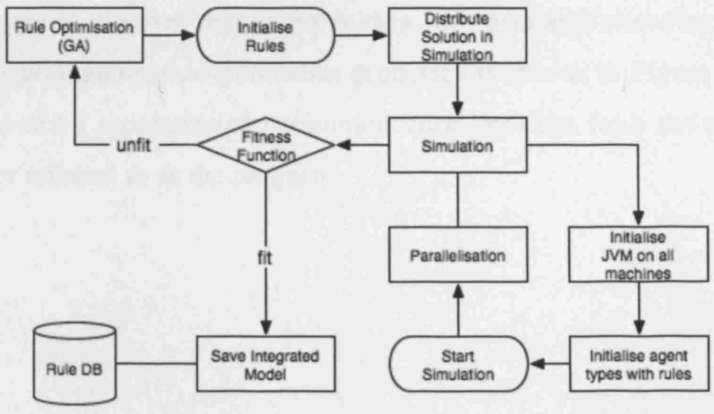


Figure 6.1 Process diagram of overall software function developed in this project.

[3] http://www.ucl.ac.uk/research-computing/services/keter/keter.html

6.3.1 Server/Client Architecture and Communication

There are two plausible architectures by which the desired application can be engineered, i.e. centralised and decentralised. In a centralised architecture the coordination of the entire simulation is handled by a single or a small number of "master" nodes whilst "slave" nodes handle the bulk of the computations. In a decentralised architecture there is no such coordination, i.e. the application runs "bottom-up" and all nodes are slaves. Whilst decentralisation is intuitively an attractive architecture to adopt given the nature of the ABI strategy it may well be more difficult to implement considering the specific parallelisation sought (see Figure 6.4). Additionally, for this proof-of-concept, an architecture in which all nodes are accessible to the point of simulation interruption and manipulation is highly desirable. Moreover the simulation initialisation process can be made far easier for the end-user if all modifications, etc., can be made from a single node. Hence a centralised structure is chosen here.

The parallelised simulation sought in this project requires an architecture by which JVMs running on different computers can deliver data generically and efficiently between each other. Java supports TCP/IP via socket classes upon which Remote Method Invocation (RMI) is based. RMI allows direct Java method invocation between multiple JVMs, albeit at a slower speed than low-level socket programming (Harold, 2004), and uses declared IP ranges to discover and communicate with available client/server computers. The overall design employed to parallelise processes is shown in Figure 6.2. A generic design is achieved by separating the communication function from the application that uses it, hereafter referred to as the recipient.

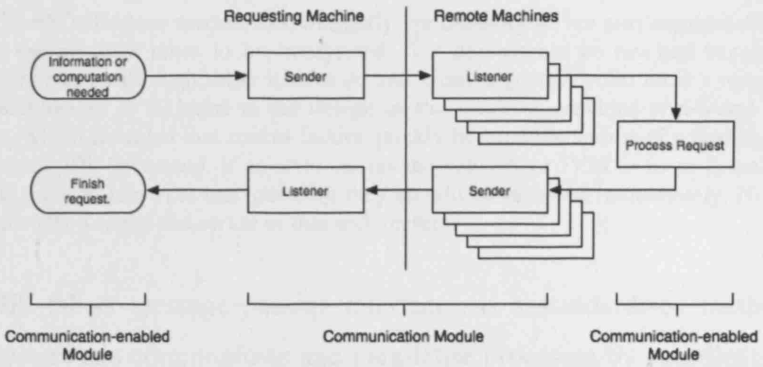


Figure 6.2 The original JVM should be able to send a message to any number of client nodes along with the instructions for computation. Coordination between the clients should use the same mechanism, i.e. non-duplicate, and a completion (solution) message should then be sent back to the original JVM.

To achieve this separation a proxy-like pattern was employed in conjunction with a delegation pattern to hide the implementation of the MPI and MPI-related classes (discussed shortly) from the recipient. `SpaceTimeModel` implementations delegate simulation requests to the communication module via the `ClientCommunicable` interface. In this way requests can be made to and from the recipient whilst the specific implementation of communication is left to a handful of custom classes.

As illustrated in Figure 6.2 the communication module uses predefined port pairs, declared in an attributes file, to send messages between client and server. Note that using TCP/IP protocol as the basis of communication provides the opportunity for the software to be equally usable over the Internet as much as it is over a private network, as presented here. The pluggable framework of classes use multithreading to elicit fast responses, i.e. keep the listening/sending port from staying in the *e* state in Figure 6.3. The length of this *e* state latency depends on the length of the message and how it is parsed.

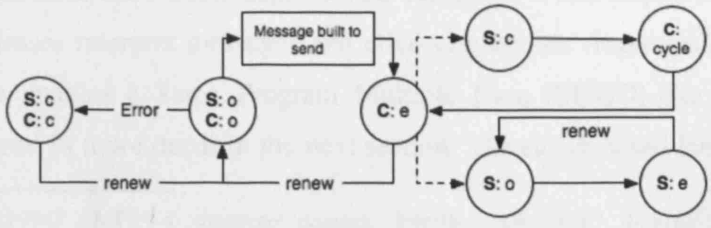


Figure 6.3 State machine diagram of port states on server (**S**) and client (**C**) sides. When both ports on either side are open (o) and a message needs to be sent (from client to server) the client temporarily

engages (e) by Java's half-close mechanism. Similarly the listening server port engages. Port renewal will take as long as the message takes to be interpreted. If a port cannot be reached because it is already engaged, the client rotates through other known communication ports (cycle) until a response is elicited. This cycling functionality is included in the design as the result of previous port-based communication implementations, which revealed that socket failure quickly became the source of a scaling problem as the volume of network traffic increased. If an error causes the network or JVM to force listening and sending ports on a single node to close (c) unexpectedly they should be renewed immediately. Note that a single node is simultaneously a client and server in this architecture.

MPI[4], or SMPI (short message passing interface), is a standardised method by which computing clusters can communicate and parallelise processes by coordination. Previous open source Java/MPI bindings on the test cluster have proved difficult to correctly function and therefore a custom MPI[5] that enables the simulation parallelisation required was developed. MPI design considerations included the need for:

- **Short message facilitation.**
  - The length of the message will affect performance.
- **Simulation initialisation information.**
  - Each node in the cluster needs to be informed as to which part of the simulation it is participating in (see §5.2.2) and how it is to be executed.
  - This was implemented as attribute files that are read at start-up and uses the same syntax as the MPI (see Appendix B2 and B4).
- **Agent information encapsulation.**
  - Positions, neighbourhood, time index.
  - States (number, string, Boolean, complex).
- **Neighbourhood requests and method invocation.**
  - Enables agents to communicate and invoke methods remotely.

Whilst the communication module handles data transmission and acquisition a decoupled set of parser classes interpret messages and elicit appropriate responses[6]. The language developed here enables a Same Program Multiple Data (SPMD)-like parallelisation, which is discussed in more detail in the next section. The string-based language not only

[4] MPI Forum (1994) "MPI: a message passing interface standard." *International Journal of Supercomputing Applications and High Performance Computing* **8**(3/4): 159-416.
[5] A full list of MPI commands is given in Appendix B2.
[6] More efficient byte stream-based commands are envisioned for future implementations.

achieves all of the requirements above but also forms the basis of a scripting language through which the fundamental aspects of the simulation, e.g. dimensionality, can be assembled and manipulated. The scripting language supports basic language constructs such as `if/else` statements and `for` loops. Complex simulation initialisations, such as those described in the next chapter, are not possible to instantiate without building new classes that describe the setup programmatically. The scripting language, in which Java-like class and method invocation is made possible by parser and executor classes that use the Java's reflection API, negates the need to rebuild and redistribute JAR files across the network (scripts are written in plain text). Moreover the executor classes are extendible, allowing future implementations to include more language features, e.g. specific commands and language constructs. A full specification of the language is detailed in Appendix B2.

6.3.2 Simulation Components

The desired breakdown of the simulation system is illustrated in Figure 6.4. Note that there are two very distinct types of parallelisable process applicable in this project: (i) the interaction-based simulation itself and (ii) executable algorithms that are otherwise serially calculated. Whilst the first is tackled directly here, the second is not as easily solved since it is dependant on the nature of the computation and is therefore left to individual cases wherein the MPI described previously can at least form part of the solution. In fact there are several extendible commands built into the prototype MPI that could facilitate distributing calculations in a cluster of participating clients[7].

[7] The one component that remains for the user to specify is the coordinator of that parallelisation in the form of a `SimControl` (explained shortly) implementation and attributes file containing an appropriate script.

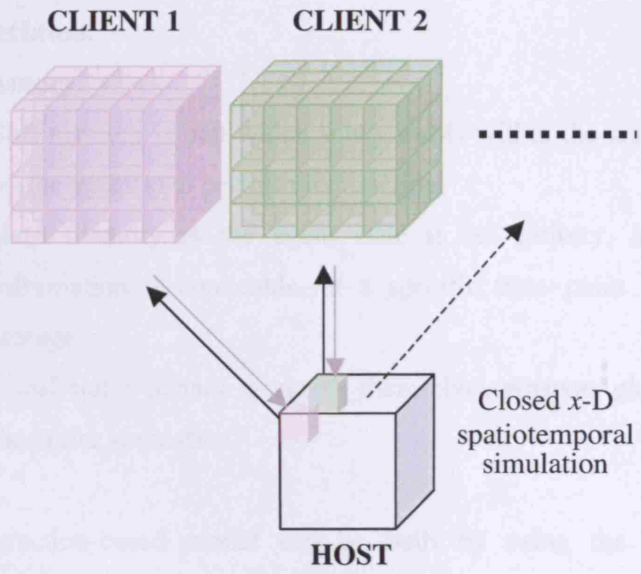


Figure 6.4 Parallelisation of spatially-oriented simulations is intuitively achieved by fragmenting the space into communicating parts (handled by the module described in the previous section) amongst participating clients. The host then coordinates time progression and any other "housekeeping" jobs whilst providing the pipeline of communication between the interfaces, i.e. the edges, of simulation parts. Ideally the host must also be able to participate simultaneously as a client to a higher-level host that performs the same function. In this way the simulation can be progressively broken down into manageable, computationally tractable parts.

Flexibility in building interaction-based systems is a difficult task to accomplish due to the sheer diversity of model types. The generic design employed here is based on the literature review presented in Chapter 2 and the reviews of Weiss (1999) and Wolfram (2002), which together cover a vast majority of fundamental components of interaction-based systems (restricted to space/time models) as illustrated in Figure 6.5.

This generic design was used as a template to create a set of classes that enables construction of most local interaction-based systems under the same software framework. Whilst these features pertain to interaction-based systems, there are a number of other components that must be integrated generically into the framework:

- **Space/time configuration.**
  - Local and global variations.
  - Time progression model (constant slice size, variant slice size, etc.).

- **Rule execution.**
- **State memory.**
    - Soft memory is that which solely exists within the simulation and is used by the agent(s) to perform its function.
    - Hard memory is the agent state in its entirety, including positional information if applicable, at a specific time point saved to persistent storage.
    - Local states pertain to agents themselves whereas global states apply to the entire simulation.

A specific interaction-based model can be built by using the basic components, implemented into a class hierarchy derived from generic interfaces (Figure 6.6), to realise the structure shown above in Figure 6.4. Behaviour relating to neighbourhood, tessellations and so on, are logically separated from other functional components by adapting Java's model-view-control (MVC) programming pattern. The `JivaContainer` classes present a view to the internal simulation model, embodied as a `SpaceTimeModel` interface implementation, which defines the space/time behaviour of the underlying interaction-based system (e.g. ABM, CA, hybrid, etc.). Since the `SpaceTimeModel` is always handled abstractly by its interface name, and therefore by its generic methods, any model can be plugged into the system in conjunction with any other model thereby enabling the implementation of complex simulations such as hybrid CA's where agents can "live" or move inside lattice sites – and lattice sites themselves can in turn implement their own space-time models, in effect creating internal simulations that can be handled by sub-clusters of computers. As such the spatial aspect of the system is independent of other components.

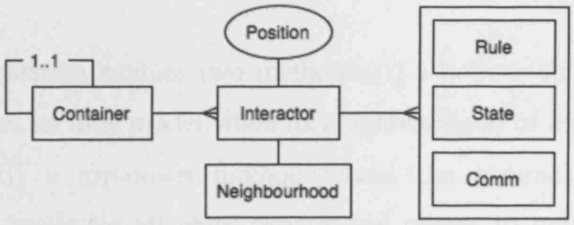


Figure 6.5 Conceptual model of an interaction-based system, centred on the interactor. All components, except for Comm, are implemented as generic interfaces through which a class hierarchy has been realised. Comm represents an interactor's ability to sense other agents and act on them, which is mediated generically as a type of communication. Both sensors and effectors are implemented via the MPI if the objects are remote or directly otherwise.

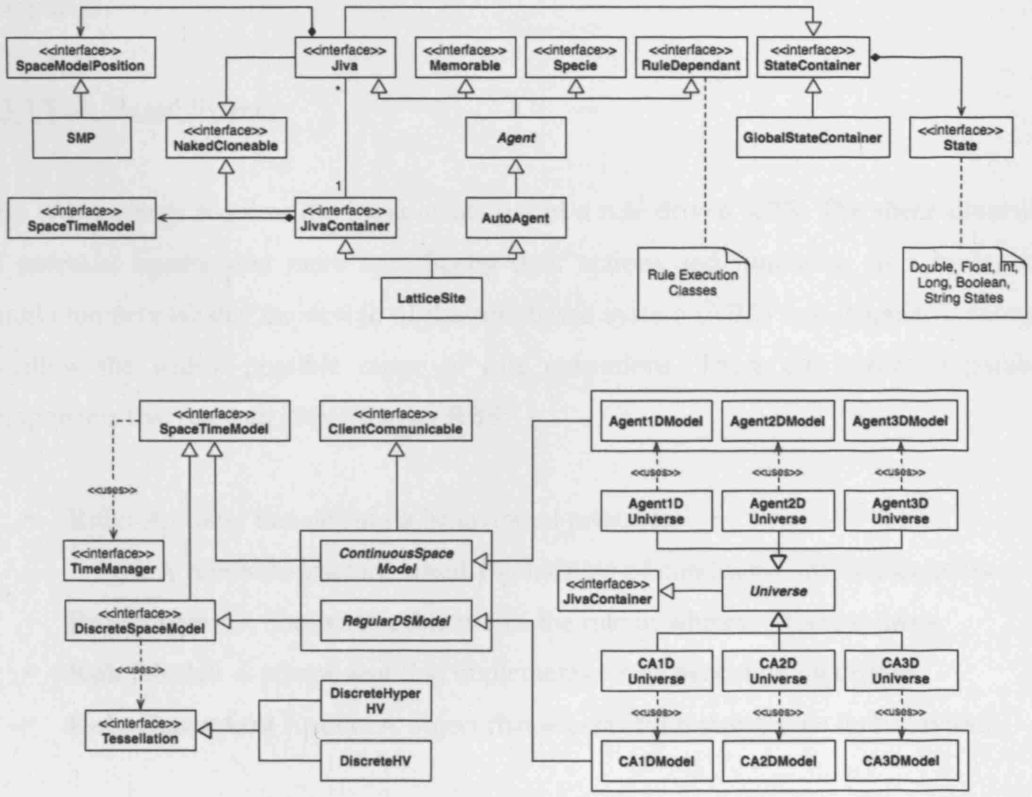


Figure 6.6 UML class diagrams of the core components of the simulation framework. This part of the project alone consists of 4 packages totalling more than 150 classes. The three most salient classes are `Jiva`[8], which represents an interactor (e.g. a CA cell or agent; top), `JivaContainer`, which represents a functional view to `Jiva` data and `SpaceTimeModel` (bottom), which handles all space and time related mathematics and runtime behaviour. In the case of discrete models tessellations are also handled by the `SpaceTimeModel`. The `ClientCommunicable` interface ensures that `SpaceTimeModels` can be plugged into the communication module described earlier and can thus communicate with other nodes taking part in the same simulation. Separating time management with the `TimeManager` class enables a choice of time strategies centred on time progression, including discrete and continuous strategies. Method and attribute details omitted for clarity.

[8] Sanskrit word meaning "individual", "soul" and "self".

The current implementation enables two methods: (i) a bottom-up method in which an interactor only updates its time model when its neighbourhood of interactors (if any) and container permit; (ii) a top-down method where the top-most simulation server increments time and waits for all child servers and clients to return before signalling another increment. Specialised `SimControl` classes coordinate the entire simulation process, including the relaying of critical progression messages and implementing specific protocols that define how data should be collected from individual client computers.

6.3.3 Rule-Based System

The ABI strategy requires the implementation of a rule-driven ABS. The sheer diversity of potential agents, and more specifically their actions and functions, in a biological simulation necessitates the design of the rule-based system (RBS) that is generic enough to allow the widest possible range of rule executions. There are several separable components that one can identify in an RBS:

- **Rule:** A clause that defines a behavioural principle.
    - A rule has structure, usually consisting of conditions and consequents.
- **Rule Parser:** A component that parses the rule in whatever form it takes.
- **Rule Model:** A component that implements a rule execution algorithm.
- **Rule-Dependant Entity:** A object that accepts rules that define its behaviour.

The RBS forms a core part of the simulation framework as illustrated in Figure 6.7.

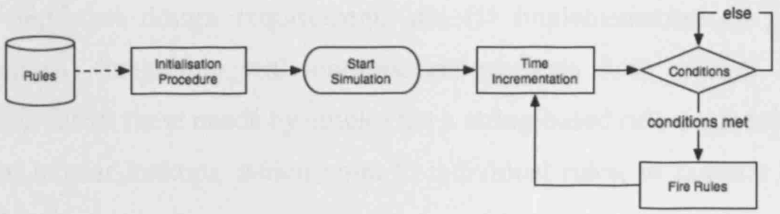


Figure 6.7 General flow diagram of simulation progression incorporating rule execution. Rules should only execute if all conditions are met. Conditions take the form of rule conditions (explained shortly) or simulation conditions, e.g. all neighbourhood `Jivas` are responding to state requests. Simulation conditions should be rechecked if they are not met. Once all rules for the current time step have been executed the simulation is free to progress. Of course time incrementation would be a concurrent process in the case of continuous time simulations.

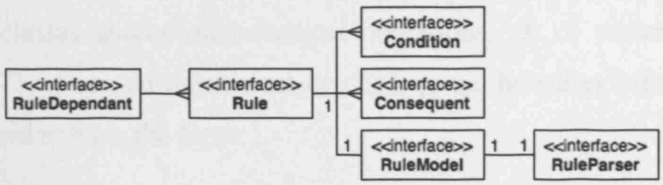


Figure 6.8 RuleDependant objects possess rules, which in turn use rule models and parsers. `Jiva` does not extend `RuleDependant` to keep the ABS and RBS components separate and generic. In the proof-of-concept application the rule models and parsers are instantiated by a factory class, which reads an attributes file to determine what model/parser to create. In this way different models can be used to change the way rules are executed and the representation format of rules can be changed without affecting other parts of the framework (in the case of rule parsers). Figure 6.9 illustrates how these components are put together.

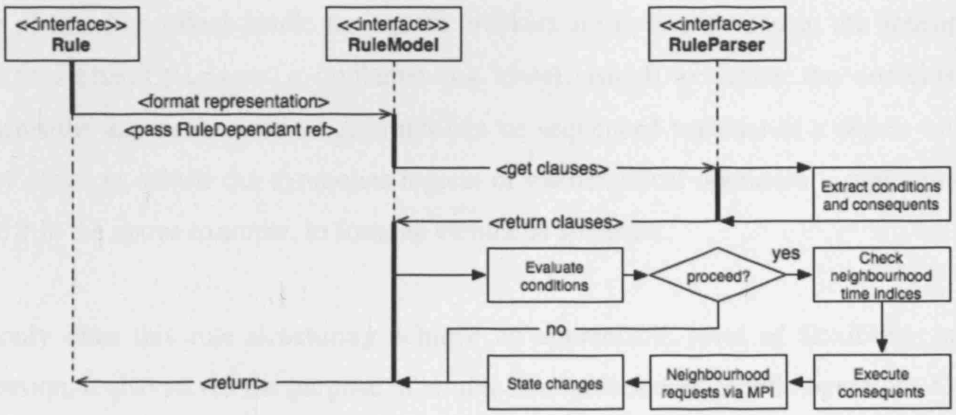


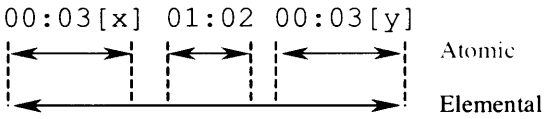
Figure 6.9 Process diagram to show the design of rule execution in this proof-of-concept.

In the software produced for this project the rule model, language and associated parser are coupled to form an extendible, plug-in-style RBS programmatic interface. Two

particularly important design requirements are (i) implementation of generic[9] rule structures and (ii) integration with optimisation methods such as GA. The current implementation fulfils these needs by employing a string-based rule engine that parses an ordered set of tabular lookups, which point to individual rules, in familiar hypothetical propositional form:

```
if (antecedent) then (consequents) else (consequents)
```

The anatomy of the antecedents and consequents can be a complex combination of predefined rule classes and/or mathematical operations, all of which are defined by lookup tables[10]. Therefore an antecedent or consequent, hereafter referred to as a rule *element*, is digitised to have the form:



where the atomic `aa:bb[]` structure contains a table lookup of value `aa` and rule set value of `bb`. The values inside the square brackets are in turn passed to the appropriate rule (an `AtomicFireable`-implementing class), which evaluates the contents and performs the appropriate action. Atomics can be sequenced together in a series with the use of adjuncts, which can symbolise logical or mathematical operators[11], represented as `01:02` in the above example, to form an elemental structure.

Not only does this rule structuring achieve an appreciable level of flexibility in rule expression, it also serves the purpose of simple manipulation of genetic operators. One of the design difficulties in creating evolutionary operators and related parsers within an

---

[9] In this case, "generic" so far as logic is concerned amounts to expressions that contain discernable condition and consequent components.

[10] A table in this case is defined as a matrix whose row and column numbers point to a secondary table, which in turn defines a secondary matrix of rule sets.

[11] Currently logical operators are only supported in the antecedent, the elemental of which yields a Boolean, whereas mathematical operators are supported in both antecedents (again, yielding a Boolean) and consequents (yielding, for example, an agent/cell state change).

RBS framework is one of rule consistency, i.e. a level of confidence that an operated rule (e.g. mutated or crossed) has the correct structure to be parsed successfully by the target rule engine. The lookup system introduced above overcomes this problem by separating classes of operators on the basis of functionality into unique tables (`aa`). For example, all mathematical operators can be put into a single table and all logical operators in another. An evolutionary algorithm that then triggers a mutation or crossover event, for example, can provide a high level of assurance that the resulting rule is of a consistent structure, which of course can greatly decrease the number of false solutions in a large-scale simulation optimisation run.

As yet there is no explicit implementation of a decision function d/μ as defined in Table 3.1 though this type of mechanism is generically embedded in a number of the said lookup tables and model-specific rules discussed in Chapter 7.

6.3.4 Interactors and States

From the generic `Jiva` interface two general-purpose interactor classes are implemented within the framework – `Agent` and `LatticeSite`. `Agents` are semi-autonomous and motile, depending only on a rule-firing mechanism for function whilst `LatticeSites` (derived from `Agent`) are static and, despite the name, are a general-purpose cell for all CA-like simulations.

States are properties of interactors and are implemented in this framework in a far more object-oriented fashion than other Java simulation engines such as MASON[12]. Obviously this has a heavy toll on memory – for a state that references a double precision decimal in optimised engines such as MASON and Desmo-J[13] the memory footprint can be as little as 16 bytes because it is simply array-based whereas in the framework presented here it will be at least 48 bytes because a state represents an entirely new instantiation[14]. In

[12] http://cs.gmu.edu/~eclab/projects/mason/
[13] http://asi-www.informatik.uni-hamburg.de/themen/sim/forschung/Simulation/Desmo-J/index.html
[14] Memory footprints determined as per Sun Microsystems specification given in http://java.sun.com/docs/books/performance/1st_edition/html/JPRAMFootprint.fm.html

simulations where there are many[15] agents that have memorable states, i.e. states with history, this toll can quickly become intractably large. The reason for employing such an object-oriented approach over simple array of single primitive/object-based references is mainly one of enabling a range of choice over the behaviour of state objects (independent from that of rules). For example, properties such as boundary conditions and arbitrary-length memory can be embedded into states and easily queried. For the proof-of-concept version this implementation will suffice but this is clearly an area that will need some optimisation.

6.3.5 Genetic Algorithm Module Development

The GA module has been developed with generic interfaces to enable flexibility in extension and modularity with respect to using alternate optimisation classes, functions and algorithms including neural networks (structure, initial biases and weights, etc.) Simultaneously the optimisation module, whatever form it takes, is designed to be coupled to the simulation framework. There are a number of ways the simulation framework can be integrated with such an optimisation scheme; Figure 6.10 illustrates two topological views of such a coupling.

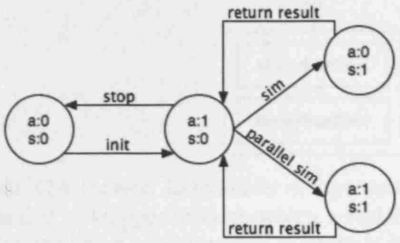


Figure 6.10 State diagram of the integration between an optimisation algorithm (a) with the simulation systems (s). Initialisation starts the algorithm (`a:1`). In a serial algorithm the simulation runs whilst the optimisation halts (`a:0, s:1`) and returns a result. In contrast a parallel algorithm continues running whilst a simulation is running (`a:0, s:1`); a possible architecture for this might entail sub-cluster arrangements wherein a simulation is spawned on a number of machines and left to run whilst the algorithm continues and spawns alternative simulations on other machines. In either case (parallel/serial) a result is returned upon which the algorithm will base its next optimisation (according to the ABI strategy described in §4.4.2). For a small cluster of nodes, as the one used to test the software, the serial option is favoured over the parallel.

[15] Typically for systems biology simulations the number is expected to be tens of thousands if not hundreds of thousands per computer (depending on the size of the simulation and cluster).

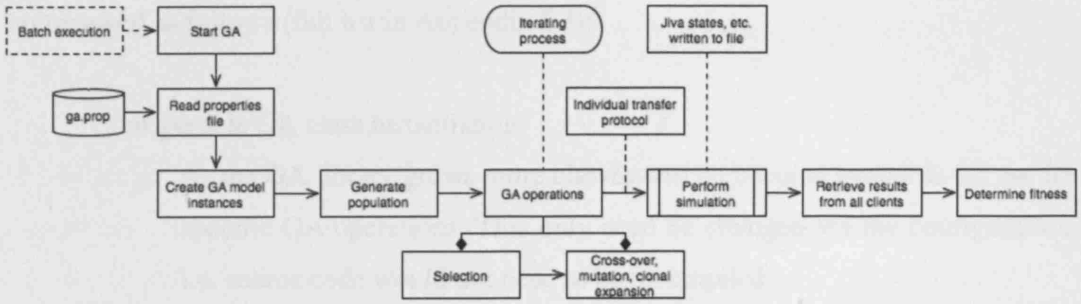


Figure 6.11 Flow of events that integrates GA rule optimisation with the simulation framework. Both frameworks are actually independent; in the proof-of-concept the property file that defines the simulation is key-value pairs coupled with scripts that delineate simulation type, e.g. dimensionality, initial values of states and Jivas, etc. Similarly a key-value file is used to define the GA, e.g. mutation rates, number of runs, generations, etc.

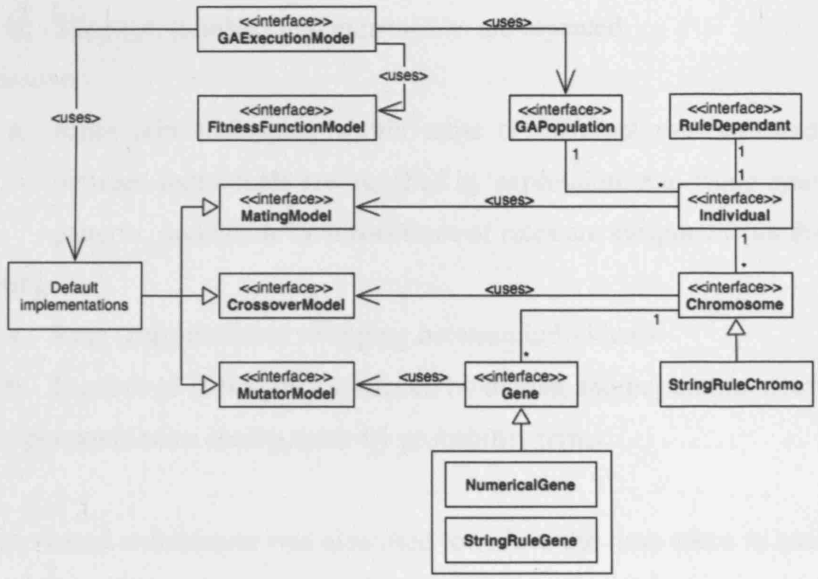


Figure 6.12 Class diagram of main GA classes. Individuals always contain chromosomes, which in turn always contain genes. An individual is a wrapper through which a RuleDependant accesses its genes (i.e. rules). The GA itself is driven by an execution model, which is attribute file-dependant, and coordinates the use of the FitnessFunctionModel, MatingModel, CrossoverModel and MutatorModel. In this design a FitnessFunctionModel must be implemented for every mathematical model that is optimised agent rules according to the ABI strategy.

The GA module builds on a highly configurable environment. An attributes file parameterises the GA and controls its general behaviour, from mutation, crossover, mating, etc. operator probabilities and their ordering to initial configuration and the

specific classes that implement operators (Figure 6.12). Controllable functionality is summarised as follows (full list in Appendix B3):

1. Configurable GA class instantiation.
    a. As the GA library grows more classes will be become available for use for specific GA operations. This only need be changed via the configuration, i.e. source code would not need to be recompiled.
2. Mutation
    a. Point mutation: numbers are randomly incremented/decremented by a configurable amount (probability distribution over defined area); like-for-like operators are swapped (mathematical and logical)
    b. Slippage: numbers and expressions are repeated.
3. Crossover
    a. Rules (chromosomes) within same or different rule sets and within or between individuals are matched by expression, e.g. *value-operator-value* patterns, and whole or subsections of rules are swapped from that point.
4. Mating
    a. Rule (chromosome) swapping between individuals.
    b. Number of individuals produced by mating and/or parental death.
5. All operators above configurable by probability terms.

This attribute-based architecture was also used to reduce the time taken to recompile and replace class files, which would potentially reside on hundreds of computers when run on large clusters. All components of the GA are designed to be pluggable, hence the separation of the GA processes shown in Figure 6.11.

The `IndividualTransferProtocol`-implementing (ITP) classes embody a complete control over how GA individuals, i.e. candidate solutions, are transferred to the target solver, which in the common case is the simulation environment and, more specifically, the appropriate interactor rule base. Since transfer is handled by a specialised class the GA can be abstracted away from the simulation environment and it therefore

can be used in conjunction with any target solver. Subsequent implementation for string-based rule enrichment only requires extension of the existing GA operator classes and the appropriate ITP.

An additional complexity is introduced in terms of the fitness function used in this context. In the case of rule enrichment, the goal is to find an optimal set of rules given initial best guesses that satisfactorily replicate the behaviour of the original model. The fitness function is dissected by defining two very separate tasks: (i) extracting output from a simulated candidate solution and (ii) comparing the output to the behaviour of the original model. The former is achieved by means of a shared translator class common to both the GA module and target solver and the latter is achieved by means of a specially-implemented model-running class that queries a model given a set of input parameters. The outputs of each can then be compared with standard statistical techniques, such as mean-square difference, to determine the overall fitness of the individual – such techniques are currently embedded within the `FitnessFunction`-implementing class. The output of the simulation itself can optionally remain distributed (if file sizes are large).

With respect to GA operation in a cluster environment and integration with the simulation modules there are two plausible architectures: (i) a centralised approach where the main GA execution is carried out on one computer, which then distributes a candidate solution for a single simulation; (ii) a distributed architecture where solutions are shared amongst a small number of interacting computers, forming a population, and candidate solutions are distributed to a number of child servers each of which can seed independent simulations. Whilst the latter is a far more powerful approach, the former is implemented currently since the size of the available test cluster is quite small.

6.3.6 Batch Execution

Since simulation times are unpredictable, especially when implementing GA or other intensive methods, batch functionality is an important tool that enables best use of

resources. Specific batch classes are created to coordinate GA and simulations serially, i.e. multiple GAs can run one after another (parallel implementations are possible but not yet fully implemented). The process, like the GA itself, is attribute file-based and it is possible to change any component in the GA or simulation setup files and rerun a defined number of times (each producing separate output files for subsequent analyses).

6.3.7 Programming Language

The choice of programming language, based on language capabilities and restrictions, available resources and ease of use, can affect the entire software engineering process. Taking into consideration the software requirements, listed in subsequent sections, ideal language features are:

- Object-oriented programming (OOP), which encourages modularity and extensible design.
- Support for distributed computing, e.g. TCP/IP-based socket programming.
- Availability of standard libraries for common tasks including:
    - I/O handling.
    - Exception handling.
    - Multithreading.
    - Collections.
    - GUI.
- Fast memory reallocation.
- Fast floating-point calculation speeds.

Secondary requirements include:

- Optional closed-source distribution, encryption and security support.
- Runtime coding, which supports ensuring greater flexibility for the user.
- Adjustable (if manual) memory reallocation to debug memory leaks, etc.

Given these requirements the natural languages of choice are C++, Java, High Performance Fortran (HPF) and Fortran 90 with MPI bindings, as well as various others.

For this project Java has been selected for a number of reasons, not least because all of the primary requirements listed above are satisfactorily fulfilled. Java is a strongly typed object-oriented language and is compiled into bytecode before deployment, which makes coding and subsequent debugging a far easier task compared to non-interpreted languages. The software produced in this project, from the outset, was envisioned to be a high performance model integration platform capable of running on a cluster(s) of computers to parallelise the entire simulation process. As such it is probable that the software may well need to run on different platforms simultaneously, which makes Java a perfect candidate since it is highly portable. The common conception regarding Java in high performance projects is that due to this portability and bytecode interpretation via the Java Virtual Machine (JVM) execution is inherently excessively slower than its C++ and Fortran counterparts. Whilst this may well be true for previous versions on older CPUs recent advancements have closed the gap significantly. Mangoine (1998) reports that JDK 1.1.5 optimised code could be as fast as corresponding C++ code and significantly faster in class loading tests[16]. Hu *et al.* (2000) report up to 70% numerical calculation performance with respect to Fortran 90 and C whereas Moriera *et al.* (2000) report up to 50% performance with the same JDK. Sun has further improved performance in Java2 5.0 using a Just-In-Time (JIT) compiler and a highly optimised virtual machine implementing HotSpot technology, which provides massive improvements in terms of speed, smaller memory footprints and better garbage collection (GC) strategies. Additionally the memory heap-size is now manually adjustable, which gives Java almost as much control over system memory allocation as lower-level languages[17], though dynamic memory management is admittedly limited. Furthermore, Java provides a number of mature mechanisms to execute native code should the need arise. Moreover Java has been the subject of many distributed and parallel computing optimisations,

[16] Floating point and integer calculations were seen to perform on par with C++ and any lower performance observations are directly be attributable to Java's inbuilt security protocol and automated garbage collection.

[17] Java 5.0 Performance Whitepaper:
http://java.sun.com/performance/reference/whitepapers/5.0_performance.html

including Sun's specific implementation called Jini[18] as well as open-source attempts at implicit/explicit parallelising compilers[19]. The latest industrial standard Java grid computing software, produced by Brain Murmurs, reports secure supercomputing up to 91% the speed of optimised and insecure C-based code[20].

6.3.8 Use of Open-Source Software

A considerable number of open-source packages exist that could constitute at least some parts of the proposed system, however such software in the opinion of the author should be used parsimoniously. Though a very attractive alternative to commercial software, the most significant disadvantage is that there is often a distinct lack of documentation and support, which is not conducive to good design within an already time-restricted project. Additionally in a project where there is specialised design goals a significant amount of recoding may well be necessary – often the time taken to understand the configuration of open-source code and adapt it is tantamount to the time taken to start afresh.

The approach taken here is therefore one of caution and most of the code is developed without the use of any third-party software.

6.4 Profiling Results and Verifications

During the development process NetBeans Profiler[21] was used to optimise code components presenting bottlenecks in performance and in particular components that exhibited memory leaks (causing `OutOfMemory` errors) and process hangs. Benchmarking was carried out on the most memory- and CPU-greedy modules of the software system to achieve a satisfactory length of execution time, i.e. to carry out the experiments detailed in this chapter. Optimisations were made to the rule engine, MPI

[18] http://www.sun.com/software/jini/
[19] http://www.extreme.indiana.edu/hpjava/
[20] http://brainmurmurs.com/products/jiva.php
[21] http://profiler.netbeans.org/

reader/writer, state-space referencing and space-time models as explained above. See Appendix B5 for all data from which figures are derived.

6.4.1 The Beowulf Cluster and JVM

Floating point operation (flop) times (on 32 bit types) were procured with the standard LINPACK benchmark[22] to verify all cluster computers were running equally well, i.e. to verify there that there would be no performance bottlenecks when the cluster was used to execute parallel simulations. The code was run in JVM/JRE 5.0 with the standard JIT compiler. The LINPACK algorithm was run 100 times on each of the 16 available nodes, results shown in Figure 6.13 below (full data available in Appendix B5). Flops speeds were calculated to deviate a maximum of 23.3% below average and 17.9% above average, however the mean deviation between all nodes was found to be 6.2%.

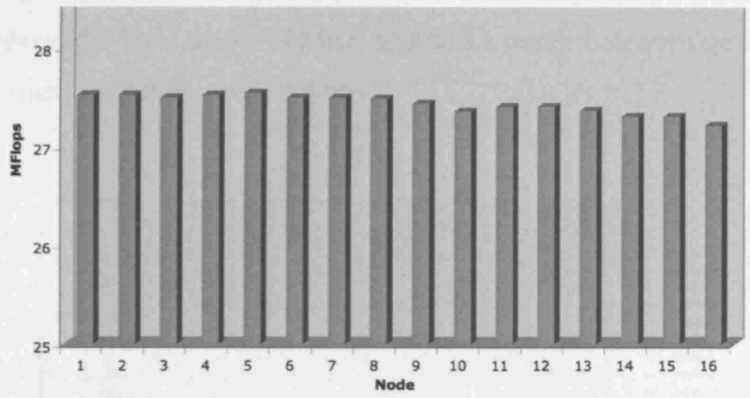


Figure 6.13 The cluster consisted of 16 32-bit Pentium III 866MHz nodes with 1GB RAM operating in the Linux RedHat 8 environment. All nodes were left running for at least 24 hours without executing any non-housekeeping processes (so as to keep swap and cache as clear as possible) before the LINPACK test was run and all system clocks were synchronised. Flops were determined simultaneously via a *cron* call to run LINPACK 100 times. Average was calculated at 27.435 MFlops (million flops) at 1.69 standard deviations.

[22] David M Doolin's (University of Tennessee) open source Java implementation of Jack Dongarra's LINPACK algorithm was used, available at
http://people.scs.fsu.edu/~burkardt/java_src/linpack_bench/linpack_bench.java

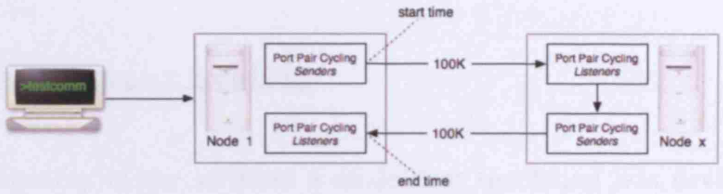


Figure 6.14 The `testcomm` command (Appendix B2) sends out a 100KB message to all clients from the master node and returns the latency in milliseconds. This was repeated 50 times and logged. An average of 8.44ms was procured from the cluster. In a separate test the time taken to traverse the logic between receiving the signal on the client side and determining it was a `testcomm` signal was less than a millisecond (confirmed using both `System.nanoTime` and NetBeans Profiler), equating a single transmission cost to 4.22ms.

Communication speeds within the cluster between the master node and client nodes using Java's TCP/IP implementation (Socket pairs) was tested by sending 100KB (represented by 102400 length byte arrays) across the network between all 15 nodes chosen serially and randomly (Node 1 was used as the master node). The test was carried out 50 times. Figure 6.14 illustrates the communication test. Though network speeds could not always be reliably sampled due to the irregularity of Linux housekeeping operations, the network switch never showed a level below 10MB/s (100MB/s max). Latency due to port cycling measured at a fraction of that speed, 8.44ms.

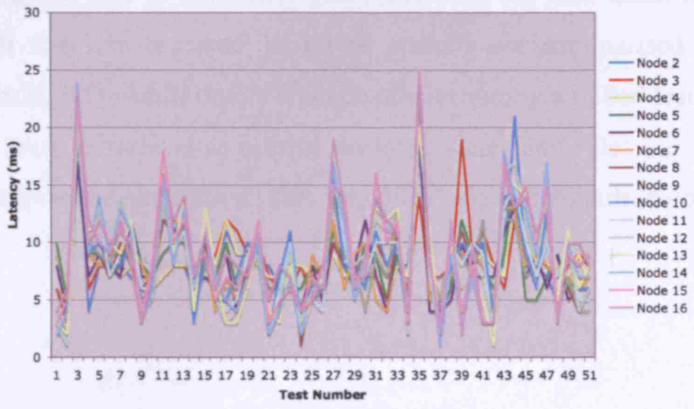


Figure 6.15 Latencies over Ethernet network. Note that `testcomm` is an asynchronous call, i.e. the loop thread does not block until a bounced signal has been received from the client (though, since it is TCP/IP-based, it will of course block until all data packets are verified as sent by the transport layer). This would explain why fluctuations in times seem to occur simultaneously as network load on the master node fluctuates.

A 4.22ms latency for a 100KB message equates to a mean network speed of 23.25 MBs^{-1}.

6.4.2 Simulation Performance Scalability

To test the scalability of the software a model test simulation was devised to run on varying number of clients and time taken to complete was logged.

A 3D diffusion solution in a 10×10×10 celled lattice was used for the test, running on a single client where a state variable (i.e. the diffusible substance) is initialised in each lattice site, and is run for 100 time steps (ts). The magnitude of the diffusible substance is set at random, between 0 and 10 units. Simulation duration was measured as the difference between the time point at which the start signal was received by the client and the time point at which the internal status-checker[23] perceived a ts=100 signal on the client machine. In each test a master node was used only to coordinate simulation initialisation whereas a varying numbers of clients were used to run the simulation.

The first scalability test consisted of a cubic lattice space simulating the diffusion model on a single node. The test was devised to measure the effect of increasing computational load by increasing the size of the lattice and measuring the time taken to complete 100 time steps. Each test was repeated 10 times. Results are summarised in Figure 6.16 (details in Appendix B5). Additionally a single client running a nulled test was performed wherein lattices were initialised as normal omitting states and rules, i.e. no calculations were made to perform the simulation. This was done as a benchmark to compare against.

[23] A threaded application that coordinates global simulation time.

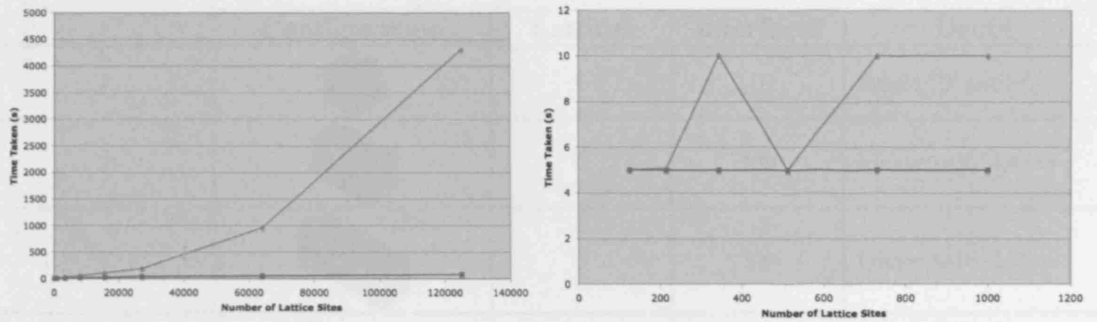


Figure 6.16 Time taken in seconds for simulation to complete with increasing size of the lattice (5^3 to 10^3, 15^3, 20^3, 30^3, 40^3, 50^3) shown in blue. The control simulation, though increasing very slightly, remains steady (shown in red). Interestingly the first six simulations, right, (5 to 10 units per side) show little difference in performance. This can be seen up until approximately 13 units length, after which performance decreases noticeably (detail not shown).

The second scalability test consisted of 6 configurations (Table 6.1) repeated 10 times, results shown in Figure 6.17. These configurations were designed primarily to test the effect of increasing interface numbers rather than just lattice size.

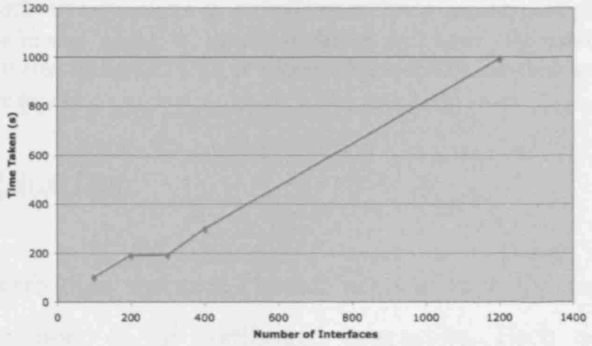


Figure 6.17 Increasing the number of interfaces has a linear effect on simulation time. A best fit curve (using linear regression function) gives a gradient of 0.814 ($R^2 = 0.9909$).

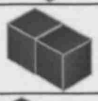
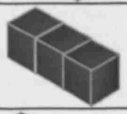
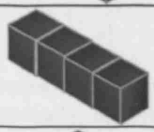
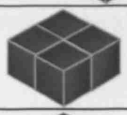
	Configuration	Lattices	Interfaces	Detail
1		1	0	Single 10^3 lattice
2		2	100	Linear 2×10^3 lattice
3		3	200	Linear 3×10^3 lattice
4		4	300	Linear 4×10^3 lattice
5		4	400	Cubic 4×10^3 lattice
6		8	1200	True cubic 8×10^3 lattice

Table 6.1 Configurations used in testing performance and scalability. Apart from the single-node scalability test, each 10×10×10 lattice runs on a single node. For example, configuration 6 would run on 8 nodes (not including the master node). A 10×10×10 lattice will have 100 interface points on each side where there is another 10×10×10 lattice. This of course affects simulation time as information needs to be exchanged between those points.

6.4.3 Logging Simulation Data

All simulations described in the next chapter need to save the outputs, i.e. the states including spatial position, of all configured interactors. Each agent in the system, including lattice sites and `Jiva` containers, keep a reference to an interval integer. When the time step is equivalent to a multiple of that number the `MemorableTranslator` object is invoked, which writes the state of the agent and its position (if not null) to disk. The overhead of writing data to disk, which is implemented as a buffered writer (i.e. the JVM invokes the operating system to write data incrementally when the CPU and disk are available) is measured by performing the standard diffusion simulation with increasing writing frequency. Figure 6.18 illustrates the results.

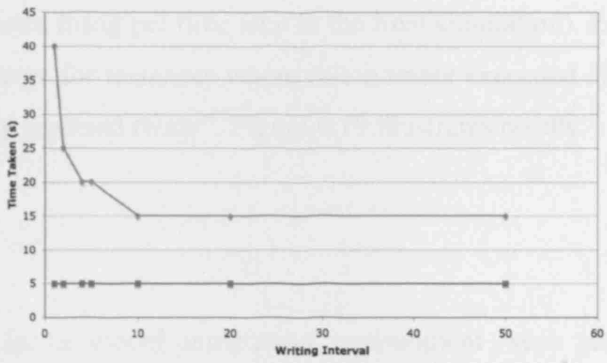


Figure 6.18 As writing frequency is increased (right to left on this graph) time taken to complete the simulation increases exponentially, but only when the frequency is below 10. This is an encouraging result as long-lived simulations would not be hindered by the overhead of writing states to disk.

6.4.4 Rule Execution

Rule execution is an integral part of the simulation software. To test the effect of rule load the standard diffusion model was executed by increasing the number of rules on the same state variables per time step.

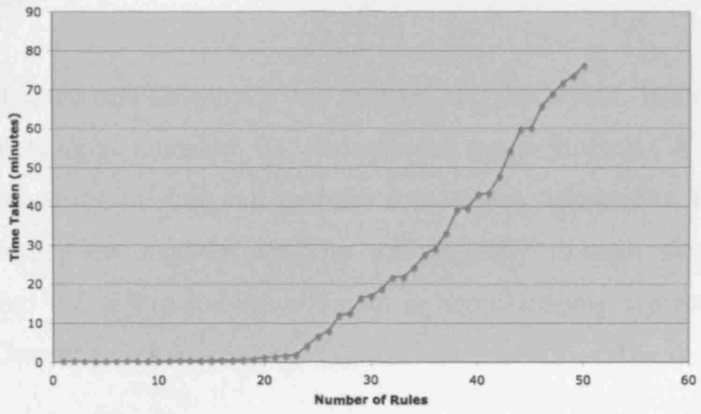


Figure 6.19 An approximately linear relationship exists between rule number and time taken to complete the simulation. However after number of rules exceeds approximately 50 (per agent, 100 agents) results become erratic – at 55 to 57 rules the time taken exceeded 2.5 hours, suggesting exponential load (not shown). The highest number of rules attempted was 100 (10^6 rules in total), which failed to cross 27 time steps even though it was left running for more than 48 hours (although it must be said that the application itself did not crash).

To avoid operating system caching skewing the result the diffusion model solution was modified so that the net "flow" (i.e. state variable changes) was multiplied by a very small but differing factor in each subsequent rule. Rule number was increased to 50 per

lattice site (50000 rules firing per time step in the final simulation). Each experiment was repeated 5 times except for instances where rule number exceeded 35, in which case the experiment was only repeated twice[24]. Figure 6.19 illustrates results.

6.5 Summary

The requirements for a model integration environment were refined to two main components: (i) a rule-enrichment module and (ii) a simulation environment. These were both implemented in Java as a multi-CPU-enabled software infrastructure and coupled to enable the KDA/ABI integration strategy described in Chapter 5. The rule enrichment module was implemented as a GA. The simulation module was developed generically to enable a wide range of individual-based simulations. These were both benchmarked to verify functionality. The software is tested with model integration case studies in the next chapter.

6.6 Conclusion

The software infrastructure developed was profiled and unit tested. These tests show that the software functions as intended, i.e. the software can produce a GA optimisation on rules and the simulation module can perform multi-agent simulations. Profiling reveals that the communication module performs satisfactorily, though there is room for improvement, and the simulation environment is approximately scalable (discussed in more detail in Chapter 8).

[24] 6 nodes were used simultaneously (and correspondingly 6 masters).

7: EXPERIMENTAL RESULTS

7.1 Chapter Objectives

The software described in the previous chapter was used to perform KDA/ABI model integrations. The design of the experiments performed including rationale is presented and followed by results procured. Chapter objectives are as follows:

- To present the simulation experiments carried out in this project to prove the efficacy of the KDA/ABI approach.
- To test the process of generating BITs to identify system/model components and behaviours (i.e. points of integration).
- To test the efficacy of using a GA as a rule optimisation engine.
- To test the process of (literature published) model decomposition via a GA from a practical perspective and subsequent integration by combining BITs and hence behaviour.

7.2 Model Integration Experiments

The aim was to validate the KDA and ABI methodologies. Note that all results and validations described below were obtained with respect to models analysed solely from publication[1]-based information (including situations where further validations required additional information from the original authors).

To validate the KDA approach 39 models were examined in detail (Appendix A1), with an extra 5 that were not in the original literature review (so as to introduce a new set of semantics that has not yet been encountered in the development of KDA) to validate the KDA process. These five are detailed below, each yielding a BIT via KDA (including

[1] Publications in the form of CellML models, such as those presented in EBI's BioModels database, could have been used instead, however this was decided against since (i) it would have required an additional layer of engineering to parse into the existing integration platform and, more importantly, (ii) semantics such as assumptions, etc., are often explained in far more detail in the original publications.

concepts that could be expressed solely from the core sets described §5.3.3, which are relatable with respect to each other) – later in the discussion this process will be justified as a technology that enables expression of Φ_I (the integrated whole). To keep in line with the cancer-oriented theme of this project only tumour models[2] were considered though it will be demonstrated that the method is equally applicable across disciplines. To test the limits of KDA, i.e. the volume of semantics that it can incorporate without yielding erroneous implications or creating practical difficulties, the models listed in Appendix A1 are used to build a large Φ_I.

Decomposition of models, i.e. the formal process *h* described in §5.3.4, was achieved by means of coupling the GA and simulation software described in Chapter 6. To simultaneously show that the ABI strategy can address the multi-paradigm problem a formal integration was performed with a new set of published models as well as with models described in the literature review. Models are selected on the basis of paradigm, focus, scope, context and assumption diversity and self-validity (i.e. the models are shown to be validated in their own right against the biological system). Additionally, for the GA to function, models were selected on the basis that a solution could be programmed into a `FitnessFunctionModel` class.

7.3 Knowledge Driven Approach: BIT-Building

In all BIT diagrams the red components represent the base classes described previously whereas green components represent those derived from the model. BIT diagrams can also become quite complex in terms of number of nodes, especially so for models that incorporate (i) a range of assumptions based on focus and (ii) many submodels, and therefore diagrams below have been condensed by categorising some concepts together. In-depth discussions follow in the next chapter.

The general heuristic used to generate the BITs from plain-text publications is as follows (Protégé 2000 was used initially to ensure the logic formalised in §5.3.3 was maintained).

[2] Except for Markus *et al.* (1999), which is a generic vessel morphogenesis model.

- Disambiguation.
    - For every component and system concept (i.e. behaviours and processes), hereafter referred to as "entry", a class is created that inherits from `SystemConcept`.
    - For every new entry three checks must applied:
        - Does the entry already exist? If so, reject the new entry.
        - Does the new entry represent a specialisation of an existing entry? If so it may be appropriate to let the new entry inherit from a more abstract one. In this case one may even granulate the model further by introducing more concepts that make the distinction between the new entry and old ones clearer.
        - If inheritance is applied, do the transitive relations break the logic as described in §5.3.3 or create any paradox? If affirmative there may be a case to go back to the original "core classes" (such cases are discussed in Chapter 8).
- Connection.
    - As far as is possible, connect each new entry with other entities using the basic relationships outlined in §5.3.3.
    - Further disambiguation can be applied with, for example, `Components` derivations that appear to be `partOf` multiple `System` derivations. In this case one should "specialise" on the `Component` in question, i.e. make them separate entries with different names but have them derive from a common parent via inheritance.
- Explicit Assumptions.
    - If guidelines are met one should see that many explicit assumptions (as listed in §4.3.1.2. Disambiguation and connection should not violate explicit assumptions, i.e. explicit assumptions take precedence over the BIT-builder's knowledge of the system.
- Initial Conditions.

- These should not be included the BIT unless they represent specific `SystemConstraints`.
- Parameterisation often can be represented as `SystemContraints`.

7.3.1 Chen *et al.* (2004)

Summary: 2D ABM model of spheroid growth with tumour, oxygen, nutrient and inhibitor states measured on 4 different layered grids.

Focus	Cell growth. Oxygen, nutrient, inhibitory agents diffusion.
Assumptions	Cell division dependant on stimulatory (oxygen) and inhibitory (generic) factors. No vascularisation. Probability distributions for entering quiescent/proliferative states.
Context	2D. States (cell level oxygen consumption, inhibitor sensitivity, diffusion-related values) each with minimum/maximums values. Varying initial conditions.

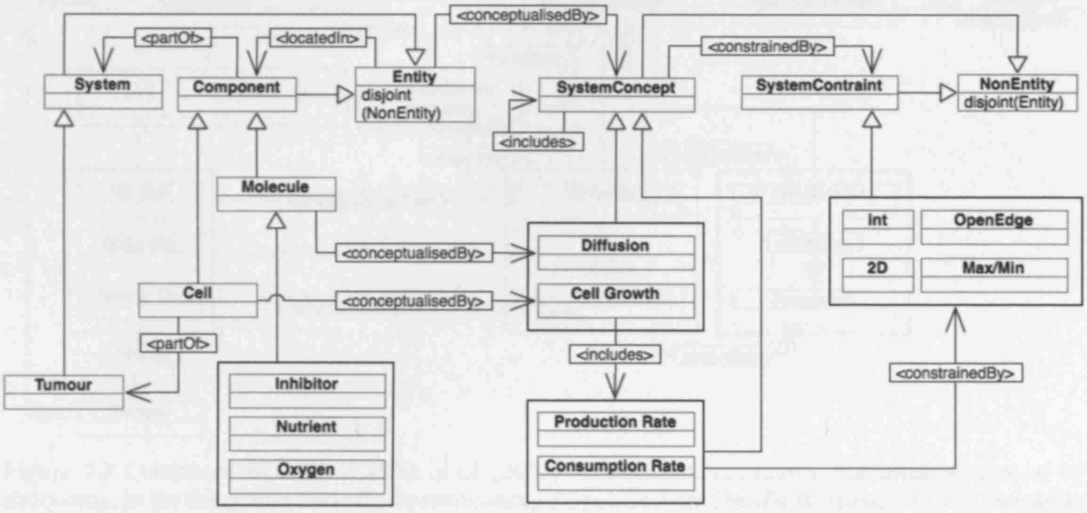


Figure 7.1 BIT for Chen *et al.* (2004). All components of the model described in the original paper can be distilled into this BIT except for logic-based assumptions that have been built into the agents (though these are implicitly included within respective system concepts), e.g. the threshold of nutrients and oxygen at which a cell makes a proliferation decision. Concepts are constrained by mathematical elements of the model, e.g. dimensionality, open boundaries (molecules can "escape from the simulation space), capped ranges for certain states.

7.3.2 De Pillis *et al.* (2005)

Summary: ODE model of tumour growth based on tumour-immune interactions.

Focus	Tumour (generic), immune (NK, CD8+) cell populations and immune cell effectiveness.
Assumptions	Growth is logistic (used as validation). Immune cell pressure (population) affects growth. NK cells always present, even in absence of tumour cells, whereas $CD8^+$ cells appear only after tumour cells have invaded normal tissue and immune cells die after a certain number of encounters. Immune cells kill tumour cells. Overall effectiveness of immune cells dependant magnitude of immune cell population, i.e. the higher the population the higher the effectiveness on tumour cells (regardless of morphology).
Context	Logistic curve variables (curve fitting to validation data). Parameterisation of ODEs (30 variables, see Table 1 in the original paper for details).

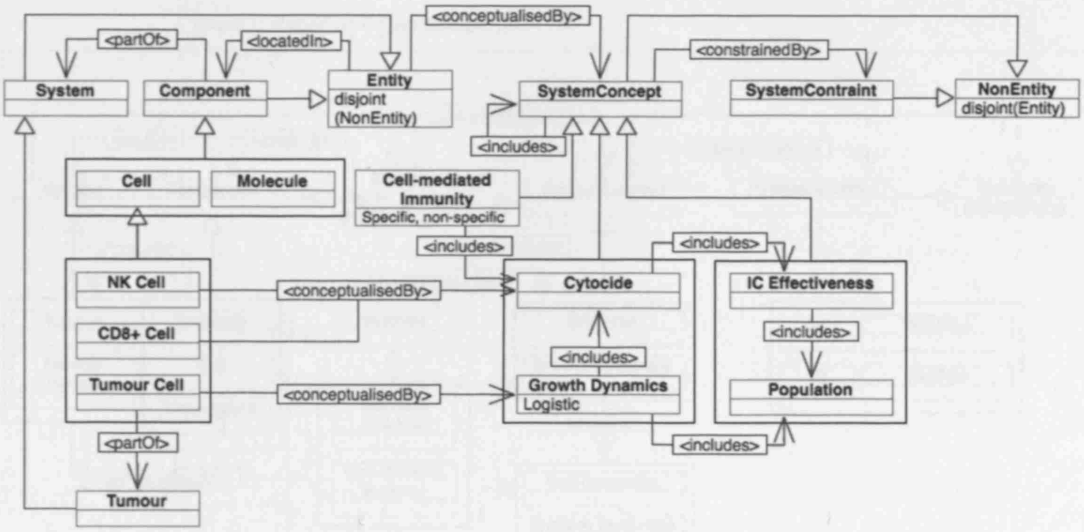


Figure 7.2 Condensed BIT for de Pillis *et al.* (2005). Components not drawn, but taken account of by abstraction, in the diagram include the SystemConcepts Specific/Non-Specific IC (Immune Cell) Dynamics and NK/$CD8^+$ Life Cycle (including recruitment and maturity). Also, some Growth Dynamics and Population have been condensed (they equally apply to all cell types and the tumour system itself). The conditional presence of $CD8^+$ cells and constant presence of NK Cells can be included by extending Population with the <*includes*> property.

7.3.3 Mallet & de Pillis (2006)

Summary: 2D CA/PDE model of tumour growth including nutrient source (abstracted blood vessel) and immune system interaction, in particular cytotoxic T-lymphocytes (CTL) and NK cells.

Focus	Host (normal), tumour and immune cell population growth. Nutrient movement.
Assumptions	Cell and molecular migration only in 4 directions. Probability distribution that determines cell fate (fixed), i.e. death by necrosis or immune cell interaction, cell division, cell migration, maximum interactions before death (for immune cells only). Survival and mitosis-stimulating nutrients treated as two different entities.
Context	Rates of consumptions and movements (diffusion coefficients) of each type of component. Probability of cell death due to necrosis or immune-cell interaction. Parameterisation of initial cell numbers and "typical" behaviours (see Table 1 in original paper). Random initialisation.

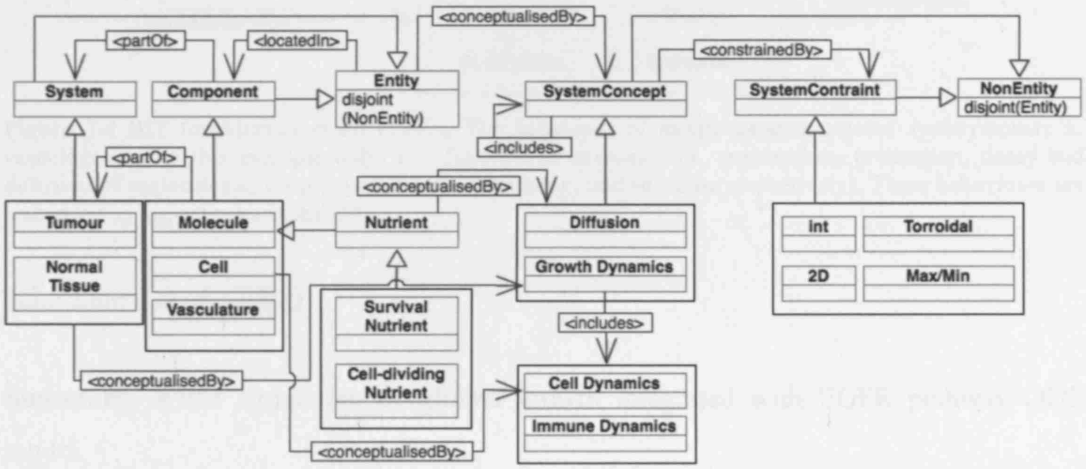


Figure 7.3 BIT for Mallet & de Pillis (2006). Much of the immune system dynamics (e.g. action of NK cells, initiation, death and proliferation) is the same as that of de Pillis *et al.* (2005) and hence condensed here into the single node Immune Dynamics. Note that due to the transitivity of inheritance and <*conceptualisedBy*> properties, subclasses of nutrient automatically inherit association to diffusion and growth dynamics, which include cell and immune dynamics (i.e. their respective behaviours).

7.3.4 Markus *et al.* (1999)

Summary: CA model of vessel morphogenesis.

Focus	Diffusion of stimulatory and inhibitory substrates.
Assumptions	Vessel formation includes growth, branching, autocatalysis of activators, tip death, chemotaxis and anastomosis, which are all driven by stimulatory and inhibitory factors only (generic). Rules favour stimulatory substrate depletion.
Context	Maximum/minimum values for substrates validated against various systems including angiogenesis and plant vein growth.

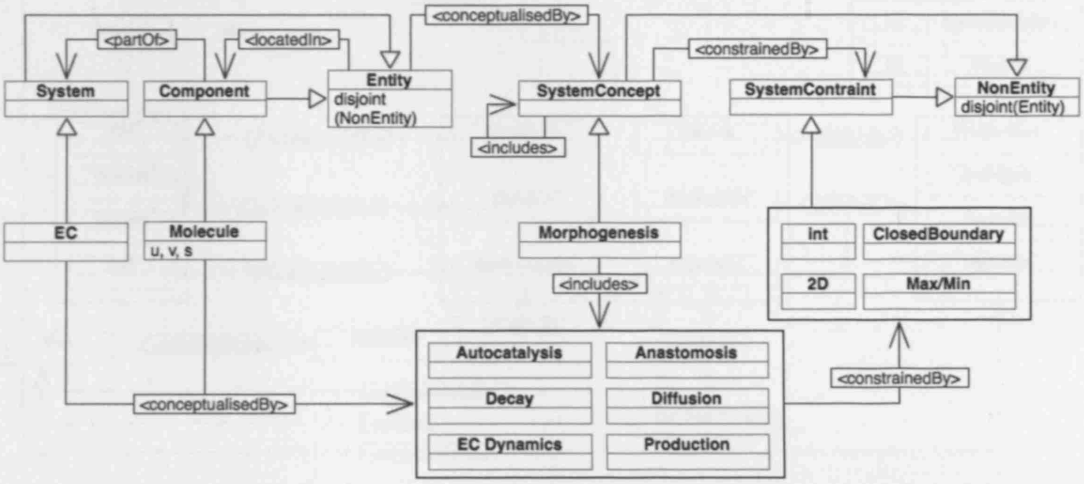


Figure 7.4 BIT for Markus *et al.* (1999). The behaviour of morphogenesis, treated synonymously to vasculogenesis in this example only, is a function of autocatalysis, anastomosis, production, decay and diffusion of molecules u, v and s (inhibitory, stimulatory and substrate respectively). These behaviours are embedded as logical rules in the CA.

7.3.5 Zhang *et al.* (2007)

Summary: ABM simulation of glioma growth integrated with EGFR pathway ODE model.

Focus	Tumour cells, chemotaxis and diffusion of TGF, glucose and oxygen. EGFR pathways conceptualised by molecular dynamics (series of ODEs), which is triggered by receptor internalisation. Main focus is the switch a tumour cell undergoes from migratory to one of proliferative, apoptotic or quiescent state (and back again).
Assumptions	Migration and proliferation dependant on EGFR pathway state. All agents occupy grid intersections – only one cell can occupy a single grid point. Cells can be in one of four states – proliferative, quiescent, apoptotic or migrating. Hypoxia and availability of

	glucose affects the state of the cell, which in turn affects distribution of cells within the glioma. Simple cell cycle dynamics are included to determine whether a cell should proliferate.
Context	Diffusion coefficients, rate constants for ODE model.

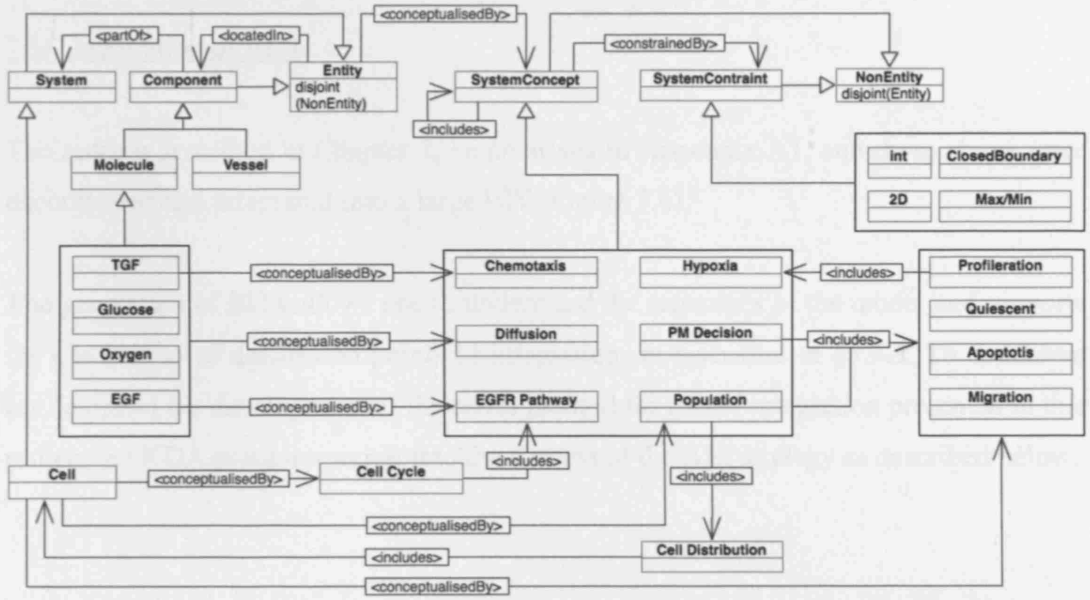
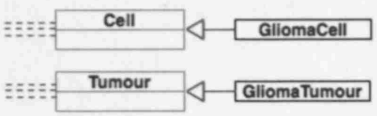


Figure 7.5 BIT diagram for Zhang *et al.* (2007). The EGFR pathway contains more behaviours than could be integrated into this diagram, e.g. internalisation of EGF and autophosphorylation (not shown for clarity). Other components can also be included that are explicitly described in the model, e.g. membrane, cytoplasm, nucleus and many molecular species but not shown to retain diagram simplicity (their inclusion can easily be incorporated as descendants of Component). Importantly, all model concepts are incorporable into the BIT diagram. The main focus of the model, i.e. proliferation/migration decision making of the cell (PM Decision above) is clearly set in context to the rest of the behaviours that the model includes.

7.3.6 Additional Notes

Cell and Tumour entities in all models described above would need to be subclassed to the specific types of cell/tumour that the models represent (to keep the diagrams as simple as possible without reducing their expressive power these child classes were left out). For example, the BIT diagram for Zhang *et al.* (2007) should include a subclass of Tumour and Cell to include the glioma types that the model was specified for:



For the most part, however, the state of the modelling literature is such that models tend to focus on generic behaviours rather than specifically on tumour types (though it must be said this is less so for pathway models).

7.3.7 Integration of BITs

The models described in Chapter 2, summarised in Appendix A1, and those above were decomposed and integrated into a large BIT (Figure 7.6).

The generation of BITs allows one to understand the semantics of the model and supports the elucidation of agents and points of integration, as discussed in §5.3.3. To document (or describe) the development of the novel method for model integration presented in this project the KDA process was put into the context of the ABI strategy as described below.

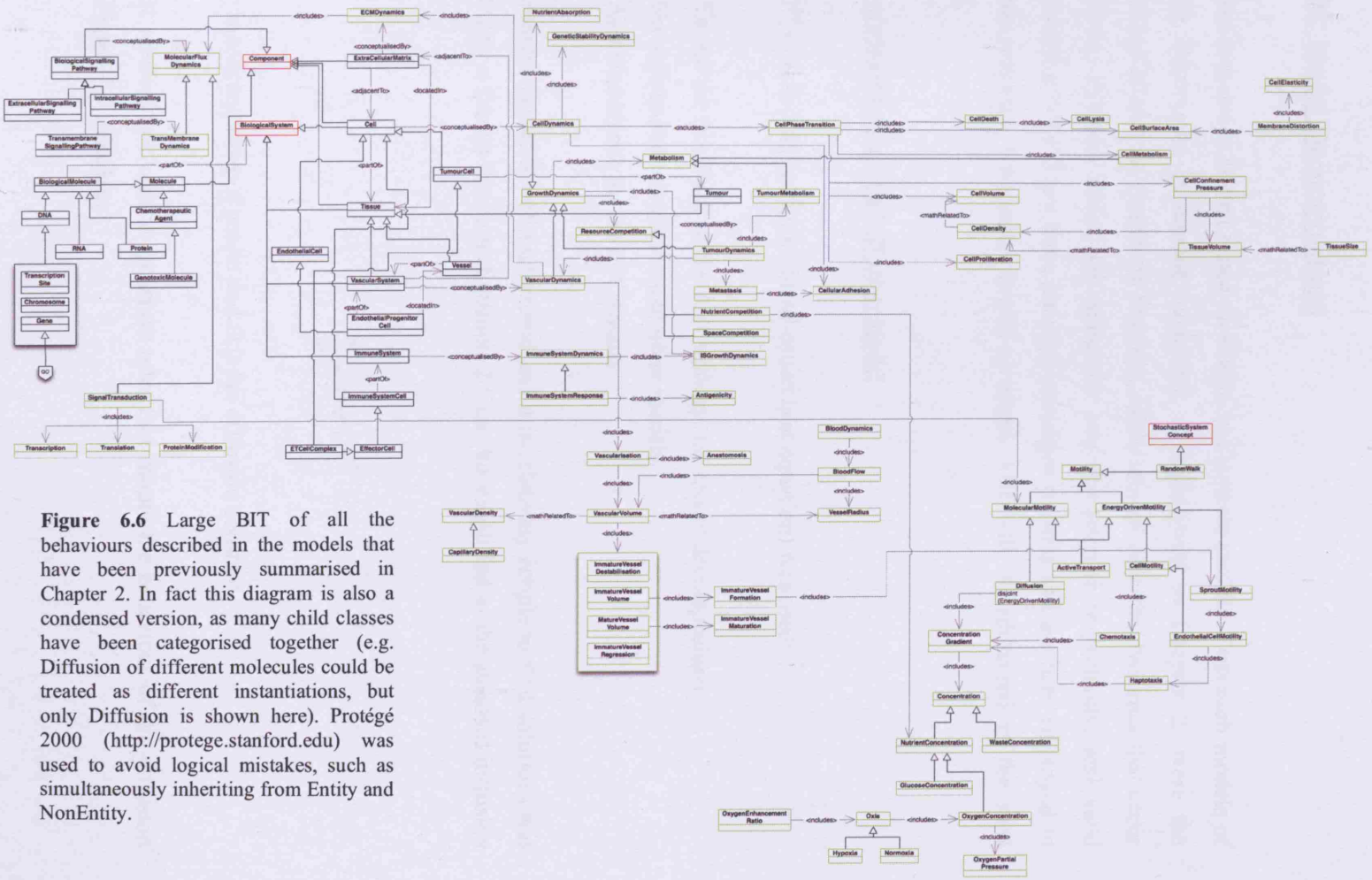


Figure 6.6 Large BIT of all the behaviours described in the models that have been previously summarised in Chapter 2. In fact this diagram is also a condensed version, as many child classes have been categorised together (e.g. Diffusion of different molecules could be treated as different instantiations, but only Diffusion is shown here). Protégé 2000 (http://protege.stanford.edu) was used to avoid logical mistakes, such as simultaneously inheriting from Entity and NonEntity.

7.4 ABI: Model Decompositions

Model decomposition is first tested on simple and testable models. Two such models of particular importance in tumour modelling, as highlighted in Chapter 2, were the diffusion model and the Gompertz function. These models describe dynamics that occur on two very different scales, i.e. molecular and macroscopic respectively, and yield straightforward BITs. They have identical paradigm profiles and are here employed to test same-paradigm integration (multi-paradigm ABI will be addressed in the next section).

7.4.1 Decomposition of the Diffusion Model

Fick's 2nd Law in 3 dimensions (the so-called heat equation) was used:

1. To test the first stage of the ABI strategy, i.e. model decomposition.
2. For subsequent integration into other models.
3. As a benchmark for the GA module.

This particular model is well understood and it is relatively simple to find solutions and hence test by a fitness function. Equation 2.5 can be simplified to the standard diffusion model:

$$\frac{\partial c}{\partial t} = D\nabla^2 c$$

where c is concentration of solute and D is the diffusion constant.

The ABI method was used to generate rules that mimic the behaviour of the diffusion model. Figure 7.7 illustrates.

Representing molecules on an individual basis is of course not a good choice in a model that intends to simulate diffusion over large spaces, as sought in a whole-tumour model, due to computational resource limits and output requirements[3]. Instead a lattice approach is adopted here wherein a discretised 3D space contains lattice sites (all of which have identical rule sets) that represent concentration(s) of solute(s) as states upon which rules operate – in effect this is a metamodelling approach to bridge the scale gap between molecular movement and macroscale behaviour akin to the Lattice-Gas approach (Versely, 2001). In the first instance 100 units per unit volume of solute was introduced at the midpoint of a 3D closed space of 10×10×10 unit dimensions and allowed to diffuse at λ=1 (denoting a zero molecular impedance)[4]. A von Neumann neighbourhood[5] was assumed on the theoretical basis that the net movement of molecules through a lattice interface is proportional to the area of that interface – this can equally be applied for higher-dimensional neighbourhoods but the von Neumann variety is relatively simple to implement (and hence relate to the said area) and is a typical and well-researched construct in discretised simulations. However the radius of the neighbourhood for which rules can operate on states was left to the GA to optimise on, as explained on the next page.

The GA was initialised with a population of 100 individuals with rules (the first generated manually, the rest randomly mutated by the `MatingModel`, `MutatorModel` and `CrossoverModel` classes before execution) and GA operator permutations to procure the results detailed below. The fitness function (explained shortly) was set to eliminate 50% of poorest performing individuals from the population in each generation. These initialisations were chosen on the basis of initialisation setups of previous GA-based rule optimisation experiments, summarised by Bossomaier *et al.* (1999) and Mitchell *et al.* (1996). GA permutations used in procuring results are summarised in Appendix C1 Table 1.

---

[3] The theoretical basis of diffusion, i.e. random movement and impedance of particles, could be encoded into an ABM, for example, as rules for individual particles of which there could easily be billions.
[4] The decomposition was later cross-validated with alternate initial conditions and diffusion constants.
[5] Explained in §3.4.1

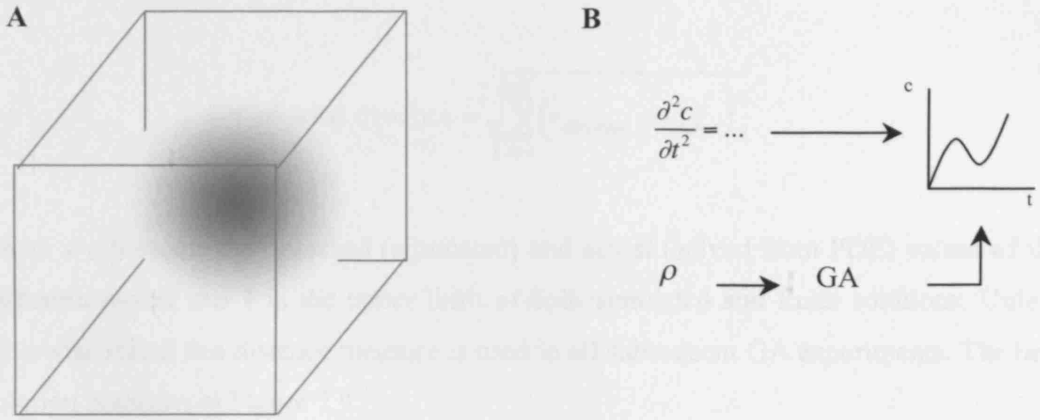


Figure 7.7. A. Introduction of a solute at a single point (or randomly distributed as a secondary initialisation, not shown) in the middle of a 3D space is allowed to diffuse in all directions. **B.** The non-dimensionalised solution of Fick's 2nd Law is used as a fitness function to enrich an initial set of automata rules such that resulting simulation mimics the original equation.

A specialised rule was hard-coded on which the GA could operate, which is detailed in pseudocode shortly. Note that the ABI strategy requires additional knowledge about the system for rule initialisation. In this case diffusion is initialised into a static rule set that operates on a state space $0 \leq s \leq 100$, which is dependant on the state space of neighbouring lattice sites – this dependency is built into the rule.

The diffusion model was separately solved to perform as a fitness function using the standard Euler's finite difference method[6] (Figure 7.8). The GA took at least 18 generations[7] (depending on initial rules and GA configuration, both explained shortly) to generate a rule set that came within the preset 0.05 distance value, measured at each time

[6] The solution required a 3D mesh on which finite differences were solved. The software used to perform the simulations in this project was used to also generate this mesh and solve the equation for various values of the diffusion constant. The result was outputted to file prior to GA execution, which was directly accessed by the fitness function model class.

[7] Best result set out of 3 repeats of various configuration changes. For full result tables please see Appendix C1.

and grid point[8] (explanation of this threshold given shortly). Fitness functions employed the standard Euclidean distance metric:

$$\text{total distance} = \sqrt{\sum_{i=0}^{i \leq T} \left(v^{i}_{Observed} - v^{i}_{Actual} \right)^2}$$

where v represents the observed (simulated) and actual (solved from PDE) values of the diffusion model and T is the upper limit of both simulated and Euler solutions. Unless otherwise stated this distance measure is used in all subsequent GA experiments. The best solution is shown in Figure 7.9.

The diffusion model was secondarily used as a test of how well the GA software component performed and functioned[9]. The first test runs of rule optimisation for the diffusion model were coded as an arbitrarily expanding rule set with the default GA configuration (Appendix C1). In summary, 100 individuals were initialised with a single chromosome and mutated prior to the GA run. The aforementioned significance threshold was first of all tested by varying between 0 and 0.1 in 0.01 intervals – initial results showed that any threshold below 0.05 rarely yielded positive results, causing a very rapid extinction of the GA population, and a threshold greater than 0.1 would generally lead to a hanging[10] population, though these results were not always reproducible. Though such a determination is of course dependant on the nature of the model that is being decomposed, a significance of 0.05 (i.e. 95% accuracy) was chosen for all subsequent experiments on the basis that it represented a good balance between yielding positive results in a relatively small number of generations without causing an unstable GA population (at least in the diffusion model case) and it is traditionally the value chosen for

[8] Note that due to resource limitation not all grid points and time points are recorded in subsequent experiments.

[9] Note, however, that this is not a primary goal of this project. The software is tested in this way purely to validate the method as a viable optimisation tool.

[10] A "completed run" here is defined as the full execution of a GA through any number of generations terminating in either a positive result or population decrease to a predefined threshold. A "hanging" population is reached when no discernable increase or decrease in performance amongst individuals is observed, resulting in GA run that never terminates, defined here as an incomplete run. Else a population becomes extinct due to removal of bad performers and lack of good ones.

biological statistical significance. In two regimes, each with one of the 2 initial rule variations listed in pseudocode below, individuals were initialised into the GA population. This was repeated at least 3 times to verify results.

1. If {true} move a fraction *f* of concentration in direction ±x, ±y, ±z over *n* surrounding sites (i.e. neighbourhood).
2. If {true} move a fraction of concentration × previous rate for this lattice site in direction ±x, ±y, ±z over *n* surrounding sites. (A variation of this rule included the addition of a random fraction of movement).

Shared-state classes (derivations of `FloatState`) ensured the conservation principle, i.e. the total concentration in the entire simulation space remained constant (excepting rounding error). Five variations on *f* in the rules above were used as initial values: 0, 0.25, 0.5, 0.75 and 1. In the case of Rule 1 all variations eventually yielded a rule set that procured the required 5% threshold, with 0.5 consistently producing the fastest result, the best of which was 18 generations. Rule 2, however, never produced a result that broke the 5% threshold – the best result was 34% after 58 generations after which there was no improvement even when allowed to run over 150 generations (spanning a runtime of more than 4 days). The addition of a random factor of concentration movement did not procure a positive result either (this was actually only performed once, producing a hanging population; results not shown).

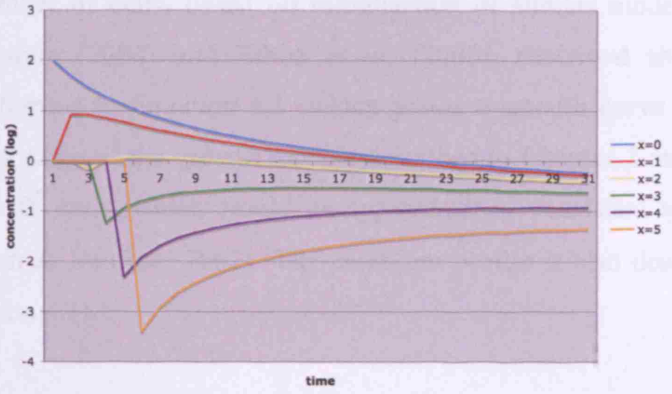


Figure 7.8. PDE solution. Log concentration of a generic solute (10 units per cubic unit of length) introduced at time 0 diffusing over a three dimensional space (only a single axis shown above, from x=0 to 5) over time. The expected exponential decrease at the point of origin is seen as the solute moves outward equally in all directions. Eventually all points in the lattice converge on a concentration of 0.01 (not shown) as the solute is completely dissolved in the available space of 10×10×10 closed-boundary space.

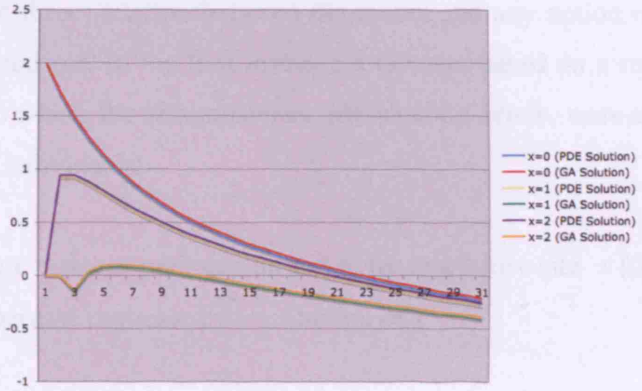


Figure 7.9. PDE and GA solutions compared at 3 points on the *x*-axis. Total distance measured as 3.69 units per unit volume.

7.4.2 Decomposition of the Gompertz Model

The same method of decomposition by GA as described above was used to translate the Gompertz model (Equation 2.4) into Λ. The fitness function employed needed an instantiation of the model to compare against. The original experiments done for decomposition of the Gompertz model revealed that initialising the simulation with a low cell number, e.g. 1 to 100, impacted simulation time and hence a single generation could take in the order of hours to complete. Therefore the initial state of the simulation was set

to a realistic number of cells, based on initialisation of similar models described by Araujo & McElwain (2004) and Ribba *et al.* (2006), described shortly. The first instantiation conformed to Equation 6.1, which yields a smooth curve very much like those procured by some of the growth models described in Chapter 2 and stabilises at a population of 6×10^5 cells, which would be approximately consistent with small MTS growths (Dormann & Deutsch, 2002). The paradigm profile is also described as the 6-tuple (Formalisation 4.1).

$$y = 5\times10^5 + 6\times10^7 \times e^{\beta}$$
$$\beta = e^{-0.005\times(t-400)}$$
$$\mathrm{P}^m_{gompertz} : \left(f^m, q^m, c^m, \neg s^m, \neg e^m, 0d^m\right) \qquad \text{Eq. 6.1}$$

Clearly there is no direct relation between the model and any notion of 3D structure or microscopic mechanisms. In the first instance a GA run based on a rate-dependant rule set was performed where the chromosomes, pseudocode below, were used to generate a population of 100 individuals:

- If {true} rate = rate at previous time step for this lattice site ± (0 •x)
- If {true} increase replicate by x ± fraction of x

Note that all mathematical operators and variable names are subject to GA operations, i.e. mutation, crossover and mating (hence the introduction of the term $0 \bullet x$ in the first rule). A total of 30 separate complete runs (and 15 more incomplete) with varying GA configurations failed to procure a result that achieved the predefined 0.05 threshold. An alternative fitness function strategy was employed where the best 25%, instead of 50%, performing individuals were progressed through each successive generation in a further 10 runs with varying GA configurations, so as to increase the selection pressure. The best performer was found to only achieve 59% accuracy.

To translate the little information available (i.e. that a variable, in this case number of cells, grows according to constants as given in Equation 6.1, which have no direct relation to actual system components) into Λ external knowledge about the system needed to be added to obtain more meaningful results (none of which are directly extractable from the original model):

- Dependent variable represents cell number (integer).
- Each cell has:
    - A point of birth.
    - A living period.
    - A point of death.
- One cell generates one daughter cell in a single proliferative cycle.
    - Single proliferative cycle defined as a varied number of steps.
- Inhibitory effect due to local population.

A more complex extrapolation can be sought wherein cell cycle phases can be incorporated however this straightforward model is used here for simplicity. The corresponding BIT is shown in Figure 7.10.

Firstly, cells were treated individually and initialised with identical rules, i.e. a child cell has the same replication rule as its parent, creating a homogeneous population. However, the cluster quickly reaches its limit after 20 to 25 proliferative cycles[11], constituting over 33 million individuals[12]. An approximation strategy was therefore devised wherein lattice sites were assigned separate lattice states representing cell number, as with the diffusion model (including the state the cells are in, i.e. living, birth/death point) – indeed this also constitutes a metamodel as described in the previous chapter. Though this proved to be more difficult with respect to constructing rules, it did eliminate the resource constraint problem completely – in fact all simulations were subsequently run on only 5 nodes (1

[11] Approximate value. As computational load increased agent and simulation logging became congested in the execution stack and so it was not clear exactly how time steps were incremented.
[12] Some threading and rule execution optimisations were made, however the JVM eventually either runs out of memory (with the upper limit on the number of threads being around 900 threads) or seizes on multiple hard-disk I/O calls (used to record output).

master, 4 slaves) satisfactorily, leaving a further two quintuplets to perform parallel GA experiments on.

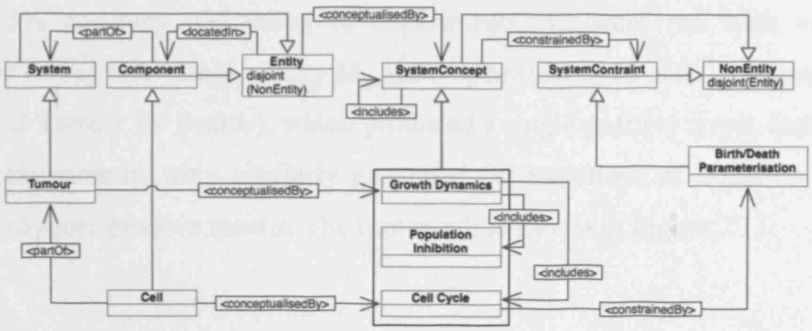


Figure 7.10. BIT diagram for the Gompertz model, with additional assumptions as given in main text.

As before, an appropriate approximation of the rules was initialised into a GA, as a starting point, consisting of 100 individuals with varying chromosome number (chromosome replication and mutation was performed prior to starting the GA to ensure a diverse population is initialised). All GA runs were made with the rule set listed in pseudocode below with varying GA and cell configurations:

Rule Set:

1. If {on death time point} die
2. If {local population $< x$} permissible = true
3. If {on birth time point & permissible} replicate
    a. Place new daughter cells randomly in local neighbourhood
4. If {true} advance point in life and cell cycle

Configurable cell attributes:

1. Length of life and cell cycle
    a. Point of birth and death set at beginning and end
    b. Number of cycles before death
2. Stochastic shortening/lengthening of cell cycle and length of life

3. Over-population threshold

Out of a total of 48 GA runs with the variations shown (each repeated 3 times) 3 reached the target 5% accuracy. Of those 48 experiments, 12 were run with varying GA initialisation (using GA values of Cr_p, M_p and Mt_p as 0.75, 0.25 and 0.5 respectively; see Appendix C2 Table 1 for details), which produced a single positive result, and a further 3 sets of 12 experiments were similarly generated for variations of Equation 6.1, which produced two more positive results. The best result is shown in Figure 7.11.

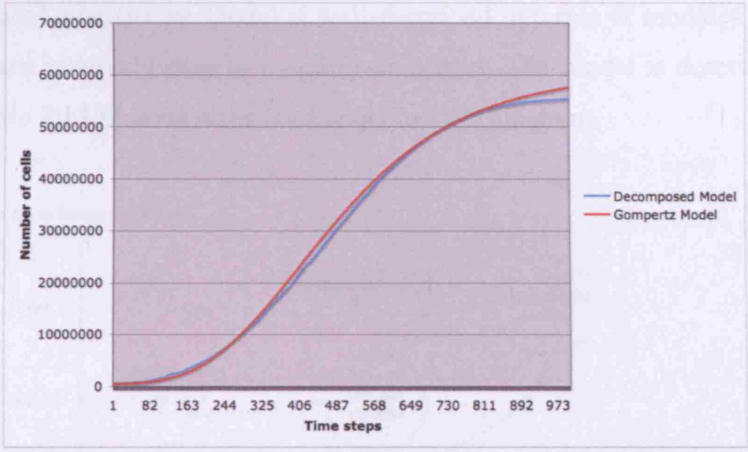


Figure 7.11. Best GA result produced a curve with a Euclidean distance of 3.3% from the actual result (see Equation 6.1). The cell cycle length was 21ts (time steps) with a life expectancy of 11 cycles and population threshold of 1253 cells per lattice site. Although after 10000ts the GA-produced model does not converge to the required higher asymptote the bulk of the curve fits the shape of the original.

7.5 KDA+ABI: Formal Model Integrations

To demonstrate the formal model integration method developed in this work KDA and ABI analysis was performed on typical tumour models to produce more informative models as per Definition 4.2.

7.5.1 Gompertz Model Integration with ODEs and Discrete Models

Ribba *et al.* (2006) describe a 2D multiscale avascular tumour model, integrating the TGF pathway and its role in cell death and proliferation into an ABM. The model is already multi-paradigm, incorporating a DE-approach to growth, diffusion and therapy response with a discrete Boolean network model describing pathway dynamics – all implemented into a discrete CA-like lattice and a parallel DE solver. This model was chosen for model integration with the Gompertz model since (i) it exhibits diverse system behaviours and therefore provides the opportunity to test how well KDA can describe a promising model, and (ii) the model is well-described in terms of modules and therefore components are relatively easy to simplify or extract. The model is described in Figure 7.12 and Figure 7.13 illustrates the corresponding BIT diagram.

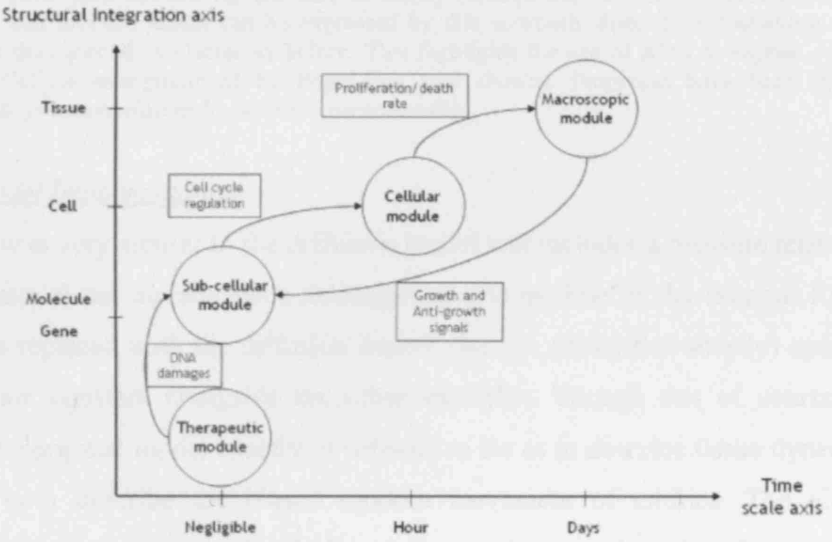


Figure 7.12. Summary of multiscale model by Ribba *et al.* (2006). A Boolean network of the TGF pathway ("Sub-cellular module"), incorporating DNA damage ("Therapeutic module"), apoptosis and proliferation, is coupled with cell cycle regulation on the cell level ("Cellular module"), which in turn in coupled with cell population pressure and diffusion dynamics in the macroscale ("Macroscopic module"). Image taken from original paper.

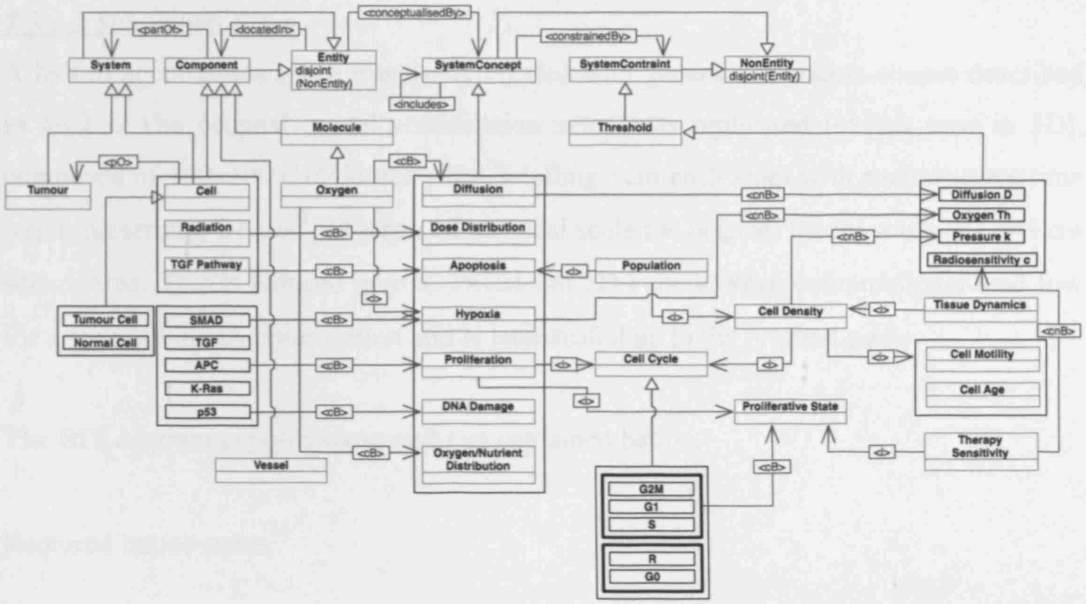


Figure 7.13. BIT diagram for Ribba *et al.* (2006). Though many of the molecular and parameterisation components have been left out for the sake of clarity (though they are indeed incorporable) the main assumptions and foci the model can be expressed by this approach. Specific instantiations are outlined black, rather than treated as classes as before. This highlights the use of KDA to expose ι as explained previously. Cell is conceptualised by Population (not shown). Properties have been abbreviated: *i* (*includes*), *cB* (*conceptualisedBy*) and *cnB* (*constrainedBy*).

7.5.1.1 Model Implementation

Darcy's law is very similar to the diffusion model and includes a pressure term. Since the diffusion model has already been decomposed this module of the original Ribba *et al.* model was replaced with the diffusion model, the GA (described shortly) optimising on the diffusion constant alongside the other variables. Though this of course does not simulate the original model exactly, it suffices so far as to describe tissue dynamics since they can both describe the biased random movement of entities. The aim of this decomposition is to integrate the ODE and discrete parts of the original avascular model with the Gompertz function, and eventually an angiogenesis model, and therefore the therapy module of the original model need not be extracted. Consequently the Boolean network was not included in the decomposition since it formed the therapy module (i.e. tumour growth after DNA damage)[13]. All other parts of the model were replicated as explained below.

[13] The possibility of further implementing the therapy module is discussed in the next chapter.

7.5.1.2 Simulation Setup

A hybrid agent/lattice space was easily created with generic simulation classes described in §6.2.2. The original model's simulation setup was replicated (except now in 3D), composed of 100×100×100 lattice sites (totalling 1cm each side) with each discrete time step representing 1 hour[14]. In terms of physical scale the original model was a 2D 8×8 cm square area. This is reduced here to 1×1×1 cm 3D cube to keep computational load low for a speedup in GA optimisation and is later scaled up to the original size.

The BIT diagram exposes states and ι as explained below.

Required lattice states:

- Oxygen level (Float).
  - Hypoxia (Boolean).
- Cell population (Integer).
  - Normal Cells.
  - Cancer Cells.
    - Cell phase population (tumour cells only).
      - G_1 population.
      - S population.
      - G_2M population.
      - G_0 population.
      - Apoptotic population.
- Central blood vessel (constant O_2 level).
  - Represented by a set of lattice sites that renew their O_2 status at every time step.

[14] Just as the `Lattice3DModel` class, defining spatial elements, can contain finer-grained spatial elements the `SpaceTimeModel` and `TimeManager` interface implementations allow for finer-grained time steps within a single global simulation time step.

Required rules:

- Oxygen diffusion.
    - Diffusion constant for oxygen.
- Cell populations update.
    - If oxygen level is below threshold or local population is over a threshold, all cells jump to G_0 (hypoxia is declared if oxygen level is below the threshold). If that is the case subsequent rules are not fired.
        - Ribba *et al.* (2006) defines overpopulation as 2000 cells aligned in the area of a single spatial unit. The lattice site is replicated here in 3D, which translates to approximately 13×13×13 cells.
- Take away fraction of O_2 from environment according to total cell number.
    - Oxygen consumption is 3.6×10^{-13} $mlh^{-1}cell^{-1}$.
- Update cycle phases.
    - Any cells at the apoptotic phase die (no nutrients are relinquished to the environment since this was not defined in the original model).
    - Increment all cell cycle steps. Reset the newly formed population from mitosis at beginning of cell cycle (G_0).

Time scale (as from the original model):

- S = 10h.
- G_2M = 3h.
- G_1 = 20h.
- G_0 = 5h.
- Apoptotic phase = 5h.

7.5.1.3 Rule Optimisation

To describe molecular dynamics and cell movement the diffusion model decomposed previously was used. This was implemented as rules that dictate the movement of cell populations (integer state) amongst lattice sites instead of a floating point concentration

as before. The original k constant for Darcy's Law ($2\times10^{-5}cm^2$) was reconstituted here as a starting point for a diffusion constant (representing media permeability) to optimise on.

The fitness function for the GA was modified to measure the scaled increase or decrease in cell population (so as to capture the shape of the curve rather than the Euclidean magnitude) since:

1. Spatial dimensions of the computational domain was increased to 3D.
2. The computational domain itself was decreased from 8mm to 1mm on each side.
3. The macroscale module was changed.

Concisely, the fitness function measured the rate of change at each time step in comparison to the previous time step and measured the Euclidean distance between observed and expected rates:

$$\text{total distance} = \sqrt{\sum_{i=0}^{i\leq T}\left(v^i_{Observed} - v^i_{Actual}\right)^2}$$

$$\text{where } v^i_x = \frac{v^i}{v^{i-1}} \qquad \text{Eq. 6.2}$$

The original model was replicated, solving the diffusion model as before on a mesh and recreating the 2D lattice with the classes described previously. As in the original model each time step represented 1h of progression. For each initialisation the states for all lattice sites and states were written to file and used by the fitness function to determine fitness of individuals.

7.5.1.4 Integration and Simulation Results

The GA quickly found a solution to oxygen diffusion and cell movement, in 8 and 11 generations respectively. Initial conditions were varied and compared to the original model to verify the rules as follows.

- Number of vessels.
- Population number of each type of cell/phase.
- Phase length.

Tumour growth according to the Ribba *et al.* model is illustrated in Figure 7.14a, with variations on cell number (initial conditions) and cell cycles of all cells synchronised. An asynchronous version of the same model, where the cells are initialised into the simulation at a random point of the cell cycle is shown (Figure 7.14b). Increasing vessel number shifts the growth curve upwards as expected (Figure 7.15).

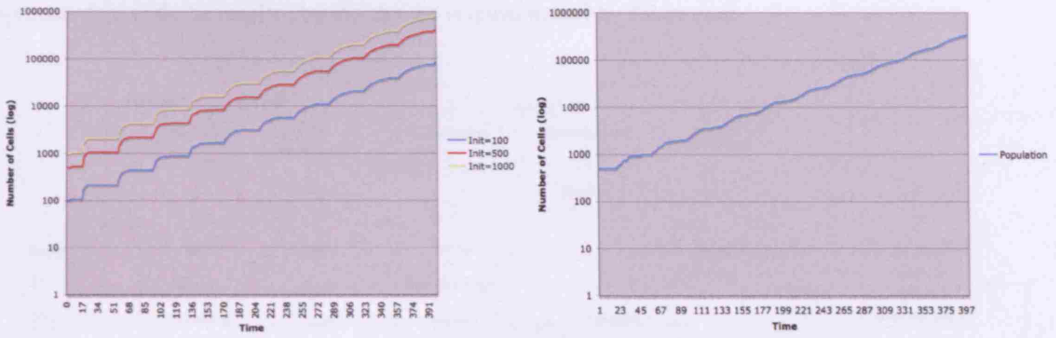


Figure 7.14 a. Log plot of cell population against time with a single vessel present. Population growth is exponential in the presence of ample oxygen. Cellular movement ensures that the majority of cells, by the time they have reached the end of G_2M phase, are clear of other cells and therefore are not affected by overcrowding until approximately t=200. Note that the original model did not specify the initialisation strategy in terms of synchronisation of cells – the effect of synchronising cell cycles can clearly be seen on the log plot, with steps of population increases at regular intervals equivalent to the cell cycle length. The effect is dissolved with the passage of time as the proportion of cells that enter G_0 increases because of overcrowding. **b.** Log plot of cell population when cell cycles are unsynchronised (starting with 500 cells). Though average population per binned number of time steps is very close to a synchronous population (compare to red curve in **a**) this curve is far smoother and the effect of overcrowding is observable earlier in the simulation.

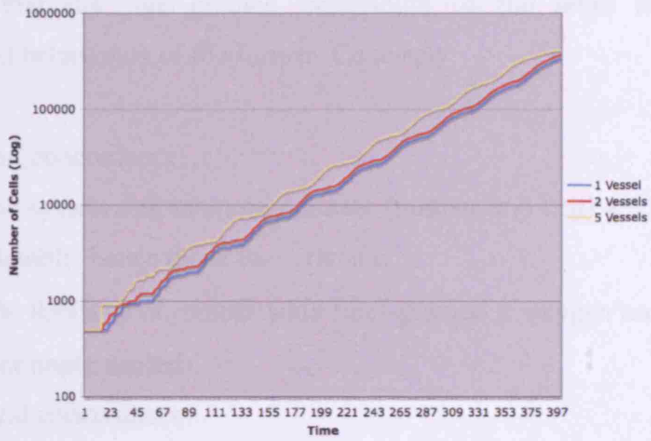


Figure 7.15 Effect of increasing vessel number shifts the curve upward. The periodic plateau of the curve is concordant with the results obtained by the original model by Ribba *et al.*

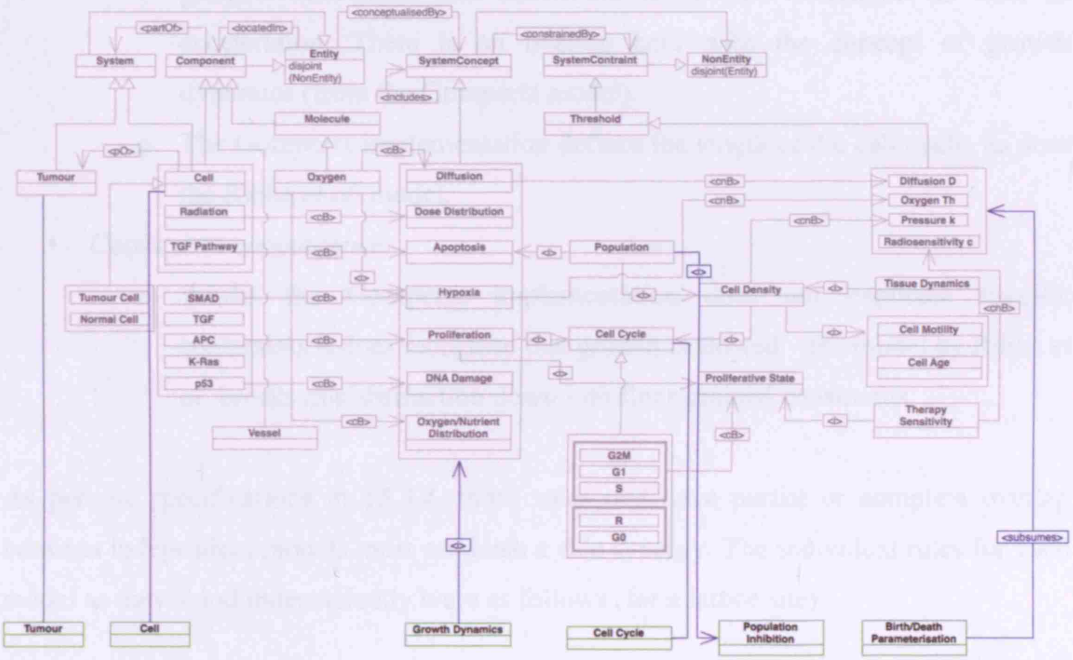


Figure 7.16. The Gompertz model is entirely phenomenological and so, as expected, its BIT integration with the Ribba *et al.* model (grey) shows that its behaviours subsume, i.e. include, the finer-grained behaviours (green to grey).

Figure 7.16 shows the BIT integration between the Ribba *et al.* model and the Gompertz derivative described in §7.4.2. Recall from §5.3.3 that BITs assert behavioural subsumption – in essence phenomenological behaviours subsume finer-grained behaviours. Indeed the integration between the Gompertz function and the Ribba *et al.*

model showed that the finer-grained behaviours of the latter are subsumed by phenomenological behaviours of the former. Concisely:

- Component concordance.
    - The system and main component (tumour and cell) are of course directly relatable, hence those two ι remain.
    - The Ribba *et al.* model adds finer-grained ι: oxygen and TGF molecules (not implemented).
- Behavioural concordance.
    - Population inhibition in the Gompertz model is directly related to the concept of population in Ribba *et al.*, which subsequently includes finer-grained behaviours that contribute to growth retardation as well as acceleration. There is an overlap here with the concept of growth dynamics (from the Gompertz model).
    - The Gompertz implementation defines the length of the cell cycle, as does the Ribba *et al.* model.
- Constraints concordance.
    - Whilst the Gompertz implementation does not explicate specific constraints it does recognise that growth is slowed – the model by Ribba *et al.* breaks this abstraction down into finer-grained constraints.

As per the specifications in §5.3.4, those rules that have partial or complete overlap between independent models must establish a rule synergy. The individual rules for each model as they stood independently were as follows (for a lattice site):

- Gompertz-derived Model
    1. If {life point = 241} die
    2. If {population in lattice site < 1253} permissible = true
    3. If {at birth-point & permissible} replicate
    4. If {true} advance point in life and cell cycle by 1
- Ribba *et al.* Model

1. If {true} diffuse oxygen
2. If {true} diffuse cell
3. If {$O_2 > 8.01\times10^{-10}$ mlcell^{-1} & population in lattice site < $(13\times13\times13)$} permissible = true
4. If {not permissible} go to G_0
5. Else
    i. If {true} oxygen uptake by cell
    ii. If {at apoptosis point} die
    iii. If {at end of cycle} replicate
    iv. If {true} advance point in life and cell cycle by 1

Using the BIT in Figure 7.16 rule integration was defined as described in §5.3.4:

Relation	Rule Combination
$\rho_R^3 \subseteq \rho_G^2$	RC2
$\rho_R^{5iii} \subseteq \rho_G^3$	RC1
$\rho_R^{5iv} \subseteq \rho_G^4$	RC1
$\rho_G^1, \rho_R^1, \rho_R^2$ and ρ_R^{5i}	RC4

where symbols G and R denote Gompertz and Ribba *et al.* models respectively, integers and Roman numerals denote the corresponding rules. Recall from §5.3.4 that, with the aim of simultaneously achieving a BIT that tends to Φ_I and attaining a synergy between different rule sets, RC4 overlaps simply add to the integrated rule set. RC1 and RC2 overlaps must perform in Λ under a belief and/or decision framework. Since a decision framework had not yet been devised or implemented at the time of execution a belief framework was utilised. The rules then became:

1. If {true} diffuse oxygen
2. If {true} diffuse cell

3. If {$O_2 > 8.01\times10^{-10}$ mlcell^{-1} & population in lattice site < [(13×13×13) or 1253]} permissible = true
4. If {not permissible} go to G_0
5. Else
    a. If {true} oxygen uptake by cell
    b. If {at apoptosis point or life point=241} die
    c. If {at end of cycle} replicate
    d. If {true} advance point in life and cell cycle by 1

Since the Gompertz model represented the most macroscopic behaviour, and due to the implications of RC3, its value of 1253 as the population limit was taken as well as a cell cycle length of 21ts. The results are shown in Figure 7.17.

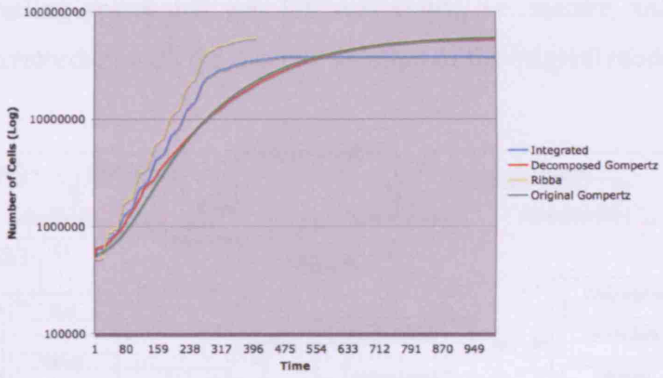


Figure 7.17. As Ribba *et al.* (2006) report in their original model the exponential growth of their *in silico* tumour does reach a point where growth slows (yellow line; population starting at 500000; curve terminates early as the original simulation only continued to 400ts). The introduction of a lower population limit and, perhaps more importantly, a death term and shortening of cell cycle length transforms the shape of the curve dramatically. The characteristic periods of oscillatory growth can still be observed. On the log plot it does seem as if the population is steadied, however a closer look at the raw data reveals that the number is actually falling with periodic spikes in growth – this may well be an artefact of the metamodelling approach (discussed in more detail in §8.3.2.2). Though the exact magnitude of the Gompertz curve cannot be achieved the similarity in shape is encouraging.

7.5.1.5 Angiogenesis Model Integration

Chapter 2 highlighted the fact that many growth models would have benefited if integrated with a vascular growth component, as this would no longer limit their applicability to MTS. Therefore the integrated model described above was further

integrated with a well-known and validated angiogenesis model first described by Anderson & Chaplain (1998) and extended by Chaplain *et al.*, (2006). Briefly, the model takes a system of PDEs to describe vascular growth, diffusion and interaction with the ECM but solves the equations on an ABM-like platform. The BIT diagram is shown in Figure 7.18.

The simulation required additional states to be assigned to lattice sites as follows:

- Fibronectin concentration.
- TAF concentration.
- EC concentration (agent count).

It was assumed that once an EC (individually modelled as an agent) had moved along a lattice site the trailing vessel that was left was viable, i.e. mature, and able to deliver oxygen. This is concordant with the discrete solution of the original model.

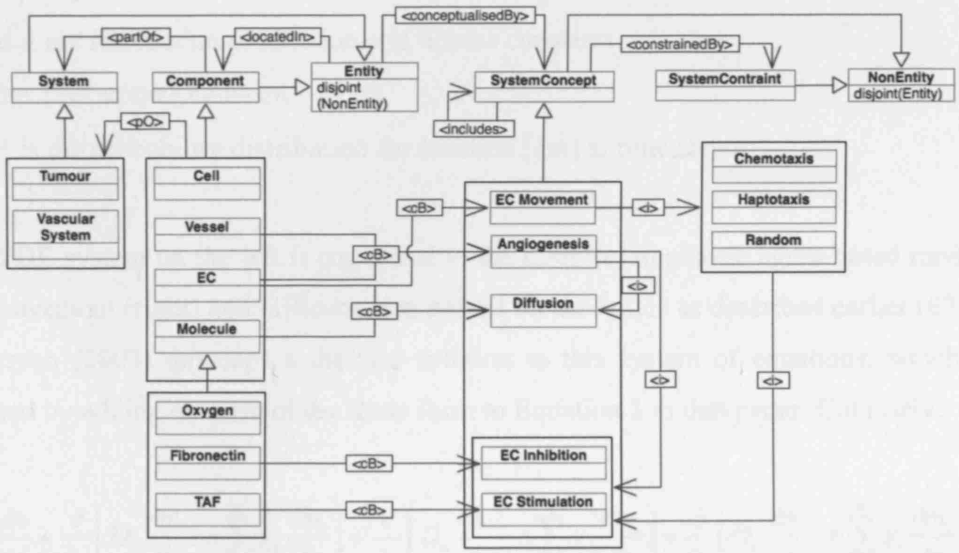


Figure 7.18. BIT diagram for Anderson & Chaplain (1998). The corresponding diagram for later versions would include terms for chemotherapeutic response, which is not considered here.

In this case the translation of the model into Λ is straightforward – Anderson (2003) described the discrete conversion of systems of PDEs describing organism migration. That solution is adopted here:

$$\left.\begin{aligned} \frac{\partial n}{\partial t} &= D_n \nabla^2 n - \nabla\left(\frac{\chi_0 k_1}{k_1 + c} n \nabla c\right) - \nabla(\rho_0 n \nabla f) \\ \frac{\partial f}{\partial t} &= \omega n - \mu n f \\ \frac{\partial c}{\partial t} &= -\lambda n c \end{aligned}\right\} \quad n_{l,m}^{t+1} = n_{l,m}^{t} P_0 + n_{l+1,m}^{t} P_1 + n_{l,m+1}^{t} P_2 + n_{l-1,m}^{t} P_3 + n_{l,m-1}^{t} P_4$$

Eq. 6.3

where:

t is time.

f and c are fibronectin and TAF concentrations.

n is EC population.

$\frac{\chi_0 k_1}{k_1 + c}$ is a chemotactic function concatenating TAF decay into TAF diffusion.

ρ_0 is the haptotactic coefficient

ω and μ are fibronectin production and uptake constants.

λ is the TAF uptake constant.

$n_{l,m}^{t} P_i$ is the probability distribution for location $[l,m]$ at time step t.

The PDE system on the left is converted to the discrete, stochastic agent-based model of EC movement (right) and diffusion was solved on the lattice as described earlier (§7.5.1). Anderson (2003) develops a discrete solution to this system of equations, which was adapted by adding a z term of the same form to Equation 3 in that paper. Concisely:

$$\frac{\partial n}{\partial t} = \frac{\partial}{\partial x}\left(D_n \frac{\partial n}{\partial x} - n \sum_{i=1}^{w} \chi_i \frac{\partial m_i}{\partial x}\right) + \frac{\partial}{\partial y}\left(D_n \frac{\partial n}{\partial y} - n \sum_{i=1}^{w} \chi_i \frac{\partial m_i}{\partial y}\right) + \frac{\partial}{\partial z}\left(D_n \frac{\partial n}{\partial z} - n \sum_{i=1}^{w} \chi_i \frac{\partial m_i}{\partial z}\right)$$

Eq. 6.4

where D_n is the diffusion constant for migration (in this case, EC), **m** is a vector of length w of molecular stimulants (in this case, TAF and fibronectin) and χ is a migration bias for each m. The probability equation becomes:

$$n_{i,j}^{q+1} = n_{k,l,m}^{t} P_0 + n_{k,l+1,m}^{t} P_1 + n_{k,l,m+1}^{t} P_2 + n_{k,l-1,m}^{t} P_3 + n_{k,l,m-1}^{t} P_4 + n_{k+1l,m}^{t} P_5 + n_{k-1,l,m}^{t} P_6$$

Eq. 6.5

In other words, P describes the probability of an EC moving in $\pm x$, $\pm y$ and $\pm z$ directions (denoted above as k, l and m respectively). Using the central difference method each P has the form:

$$P_r = \frac{kD}{h^2} - \frac{k\gamma}{4h^2}\left(m_{position}^{q}\right) \qquad \text{Eq. 6.6}$$

The 3D grid on which this solution operates is the same as Λ therefore translation into the simulation is direct. The two modifications made here are to simulate to the angiogenesis rules in hours rather than the 1.5 day time step used in the original model and to use a 3D version of the stochastic equation given on the right in Equation 6.5. Diffusion of TAF was implemented by reusing the diffusion model solved earlier (§7.4.1).

In the original model it was assumed that TAF concentration gradient was constant, using an initialisation that had a downward gradient from cells to vessels and the opposite gradient for fibronectin. This setup was replicated here simply by presetting the lattice site states before starting the simulation but was simplified in that it was linear rather than sigmoidal. This was merged with the setup described for the Ribba *et al.* decomposition except that the parental vessel is moved to the top (i.e. y=0) and ECs allowed to migrate from there (initially 1 cell density).

The integration between the Ribba/Gompertz model and the angiogenesis yielded the BIT shown in Figure 7.19.

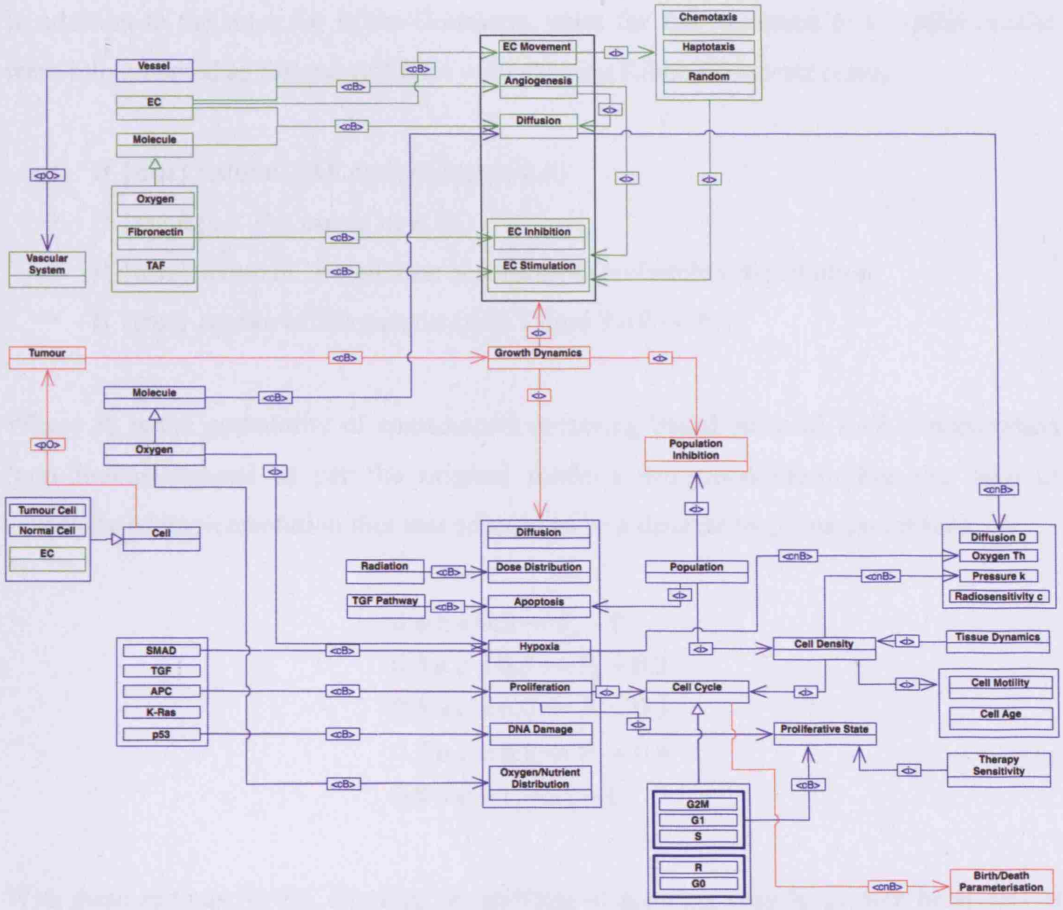


Figure 7.19. BIT diagram of Ribba (blue), Gompertz (red), Anderson (green) integrated model.

Novel ι and states added to Λ were determined:

- Endothelial Cell.
  - Implemented as an individual agent within a lattice site.
- Vessel.
  - Implemented as a Boolean state of a lattice site where the state is set to true if an EC traverses through it (this is an extra embedded rule for the EC agent, not shown in pseudocode above).
- TAF and Fibronectin.
  - Implemented as float states for each all lattice site.

In addition to the rules for Ribba/Gompertz, rules for the Anderson & Chaplain model were implemented as follows (all RC4 with existing Ribba/Gompertz rules):

- If {true} natural TAF decay (decay=0.6)
- If {random < P_b} create new EC
- If {true} move EC population according to probability distribution
- If {true} uptake of fibronectin (μ=0.1) and TAF (γ=0.1)

Where P_b is the probability of anastomosis occurring based on local TAF concentration (non-dimensionalised as per the original model's implementation). For the sake of simplicity of implementation this was adapted to be a discrete step function where:

$$0 \le c < 0.3 \rightarrow P_b = 0$$
$$0.3 \le c < 0.5 \rightarrow P_b = 0.2$$
$$0.5 \le c < 0.7 \rightarrow P_b = 0.3$$
$$0.7 \le c < 0.8 \rightarrow P_b = 0.4$$
$$0.8 \le c < 1 \rightarrow P_b = 1$$

With these settings for EC cloning, i.e. splitting of growing vessels, growth behaviour is quite similar to the Ribba *et al.* curve except stretched over a longer time period (Figure 7.20). Growth is exponential as oxygen supply to cells becomes ever more available. However growth is capped as lattice sites reach maximum capacity.

As the ECs reached the other end of the simulation space, where $c > 0.8$, the number of EC duplications per time step increased exponentially. Even so maximum cell number did not reach more than 7×10^7. This is because despite the availability of nutrient the population constraint and death equalise any extra growth – a larger simulation space would of course remedy this. Growth of vessels in shown in Figures 7.21 and 7.22.

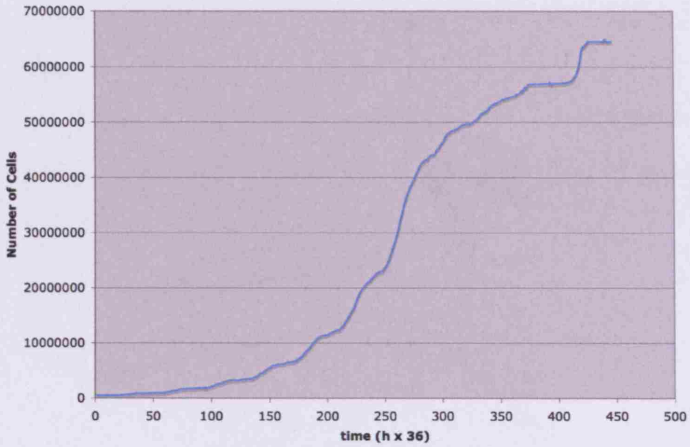


Figure 7.20. Growth is erratic as the slower time-steps for angiogenesis progress and deliver oxygen to cells. The shape of the curve is very similar to previous growth curves, however since the tumour is now placed on the side of the simulation space rather than the middle growth appears to be a little more oscillatory since the cells are not free to divide and invade equally in all directions.

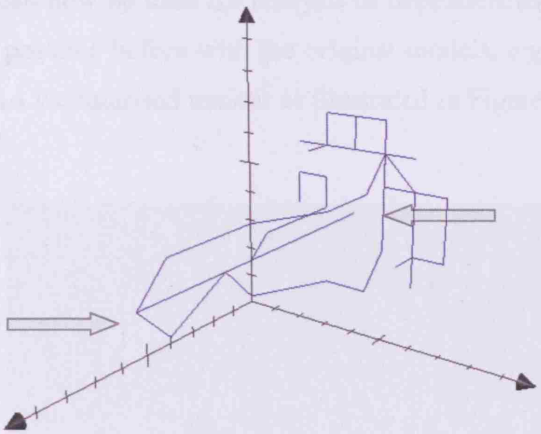


Figure 7.21. The initial tumour seed is planted (right arrow) at the opposite end of the EC seed (left, arrow). The linear TAF gradient plane ensures a steady directional growth of vessel (blue lines) to tumour site, carrying oxygen with it.

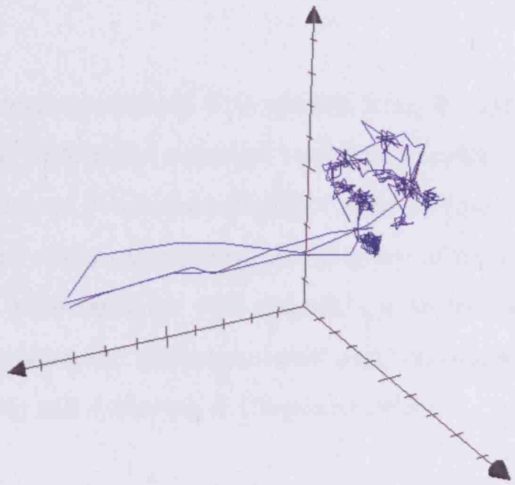


Figure 7.22. As the tumour grows outward the vessels back up on themselves (only a small set of separate vessel paths are shown here for clarity). Eventually EC movement becomes entirely random as the TAF bias depletes due to degradation.

The integrated model can now be used for analysis of dependencies between parameters *in silico* that were not possible before with the original models, e.g. the distribution and flux of hypoxic cells in a vascularised tumour as illustrated in Figure 7.23.

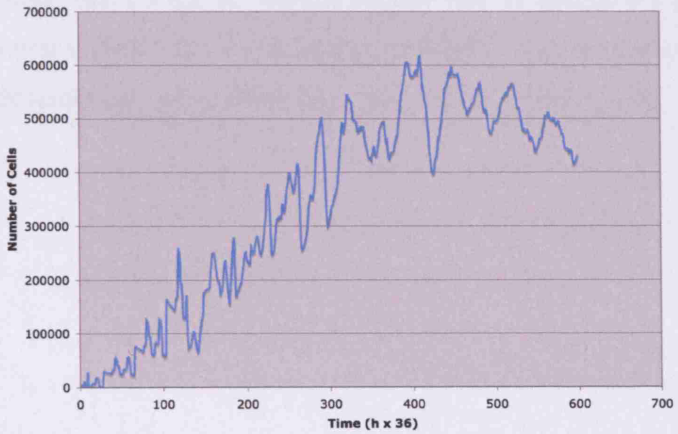


Figure 7.23. Fluctuations in hypoxia, i.e. total number of cells in a lattice site whose hypoxia state is set to true, whilst vessels grow.

7.6 Summary

KDA and ABI have been demonstrated with models from the literature. KDA has been shown to be able to cope with model concepts very well and due to the generic nature of integrative properties, such as <*conceptualisedBy*> and <*includes*>, BITs from different models can be interlinked whilst maintaining the integrity of the original BITs. This was shown by performing a KDA analysis with 5 models from the literature. ABI was first demonstrated by decomposing the diffusion model and subsequently with the Gompertz model, Ribba *et al.* (2006) and Anderson & Chaplain (1998).

7.7 Conclusion

The ABI strategy is able to decompose models into interaction rules, using BITs to reveal behaviours and components, and hence enabling model integration. Admittedly more experimentation is needed since the KDA/ABI method seems to be capable of more, however for this proof-of-concept the simulations done here suffice to show the methods do work. Critically, agent rules derived from different models can be combined in such a way within an ABS framework to yield a model that is qualitatively similar to the individual components. Since the formalisms, granularity and semantics of the original models are so different an integration like this would otherwise be very difficult to achieve.

8: DISCUSSION

8.1 Chapter Objectives

- To critically analyse the ABI and KDA as individual, as well as combined technologies, from both theoretical and practical perspectives.
- To analyse the efficacy of GA as an optimisation engine.
- To define and discuss the major benefits and drawbacks of the techniques developed in this project.
- To discuss future implementations and improvements.

8.2 Brief

The ABI and KDA theories detailed previously are critically analysed in this chapter in light of the simulations performed: to what extent can KDA describe a model? To what extent can the KDA/ABI strategy enable model integration? Do these developments make any significant advance on current technology? In each case the advantages and disadvantages of both KDA and ABI are discussed. All of these points will be discussed and will be put in the context of the current state of the cancer systems biology field. Whilst the novel model integration strategy is the focus of this thesis the software produced to make model integration and experimentation possible will also be discussed. The questions that will be asked will revolve around whether design requirements were met satisfactorily and where improvements can be made.

8.3 KDA/ABI Performance

8.3.1 The Knowledge-Driven Approach

Concisely, KDA was developed to describe model semantics, i.e. model focus and scope, in a way that was computable (i.e. expressible in a language that can be interpreted

computationally and operated on – for example, possibility of scope processing as per Definition 4.13) and in which the process of model integration would be made more explicit by means of providing a computational common ground (Definition 5.1). This is parallel with the idea of using local interactivity in ABI as the common ground – KDA recognises that the commonality between all models, regardless of focus, scope, scale or paradigm, is that they all relate a set of system concepts, specifically *behaviours*, to a set of foci. This is indeed at least true for the cancer modelling field:

- There is always a focus, i.e. the tumour or related subsystems/components.
- A focus is always attributed with state changes, i.e. behaviour (see Definition 3.2).
- Though not always explicit, there is a persistent notion of associations between behaviours, e.g. inclusion or exclusion of known processes or black-boxing.

Hence the generic classes and properties illustrated in Figure 5.1 were distilled from an in-depth literature review. Note the great advantage of the generic nature of KDA is that its specification is not restricted to biological systems but any system that conforms to the structure and definition of system as described in §3.2.

8.3.1.1 Case Studies

The 5 models that have been successfully described in detail with KDA in §7.3 illustrate the point that there are actually very few concepts about a model that cannot be described by KDA's basic classes (or extensions thereof). One case that was not tested in this project was the non-functionality[1] paradigm. However even in this case, though behaviour and hence behavioural inclusion are not an inherent property of the model, the KDA technique still affords the possibility of describing the model on the basis of its connection with the real system, i.e. its focus and relationship between foci. The few concepts that could not be covered by KDA are as follows:

[1] This was due mostly to the fact that there are very few non-functional models in the cancer modelling literature if any at all.

- Chen *et al.* (2004)
    - Logical rules pertaining to how oxygen and nutrients are replenished at the edge of simulation space could be interpreted as assumptions. KDA can only take this as far as those concepts exist, i.e. Oxygen and Nutrient Production, Consumption and Diffusion concepts. However these also can be interpreted as a property of the solver and hence have no place in KDA – they do not add any knowledge of the foci and how behaviours are related with each other.
- de Pillis *et al.* (2005)
    - None
- Mallet & de Pillis (2006)
    - Assumption relating to the equivalence of diffusion constants, to keep in line with an earlier model on which this is based. This was generically inputted into the BIT as a Diffusion concept (for each diffusible molecule).
- Markus *et al.* (1999)
    - None
- Zhang *et al.* (2007)
    - None

A consistent consideration whilst building these BITs was whether initial conditions should have been included. Initial conditions do indeed affect behaviour of the model and should be in the inclusion tree (or perhaps as `SystemConstraints`) on that basis. However it can be argued that they are also part of the solver, from which KDA is supposed to be abstracting away and so were not included. This decision was also based on the fact that the process of model integration was not contingent on any assumptions related to initial conditions. Indeed if the process of decomposition into rules is successful the initial condition setup should be replicable within Λ.

8.3.1.2 Evaluation

Based on these decompositions, as well as the 39 summarised in Appendix A1, it can be concluded that the core requirements outlined in §5.3.3 have been fulfilled:

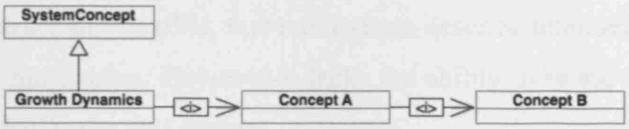
1. *High level of capture of the semantics of models.*
    - As described above, there are very few concepts that are not immediately translatable into a BIT. In fact, not only can this approach describe almost all model concepts, it often exposes relationships that are not made explicit in the model, e.g. the chain of behavioural inclusion.
2. *Direct and dynamic mapping of actual system components.*
    - *The representation should fully reflect the original modeller's knowledge of the system, i.e. what is termed "expert knowledge" in the literature.*
        - KDA requires the user to build relationships between systems concepts and system components. Much of this will come from the model itself. However integration of BITs requires the user's knowledge of the system to expose behavioural inclusions and equivalence as well as disambiguating system components.
    - *The representation should be robust to change in or introduction of new knowledge.*
        - There was not a single case in any of the integrations of BITs that required a change in properties or classes (i.e. the structure and therefore semantics of the BIT).
        - A single implementation is robust to change insofar as rearrangement of elements is concerned. However an integrated BIT is hypothetically more difficult to manage in that once a major change is needed one may need to go back to the integration (ABI) stage. That said, however, a change in KDA (i.e. belief in the way the system is behaving) would constitute a change in the model and so reiteration into ABI would be necessary anyway.
3. *High level of expression with a relatively simple representation to avoid error.*

- *Flexibility in what can actually be represented is vital as complex models of complex systems will presumably go hand-in-hand with complex semantics. Moreover these semantics may well incur human error when the (graphical or exchange language) representation is difficult to understand or generate.*
    - Though BITs become progressively complex and difficult to draw neatly one must weigh this minor disadvantage against what the diagrams represent. Even for simple models there is a wealth of semantics that must be described and the BIT diagrams manage to do this using generic components in a relatively straightforward fashion. Moreover it would seem that there is a point of "critical mass" where the proportion of concepts in the BIT actually starts to fall as more models are added to it since the existing BIT already contains many of the behaviours already – this is clearly seen in Figure 7.6 which contained 104 `SystemConcepts` from 39 models (ratio 2.67) whereas the integrated Ribba-Gompertz-Anderson model contained 48 `SystemConcepts` (ratio 16).

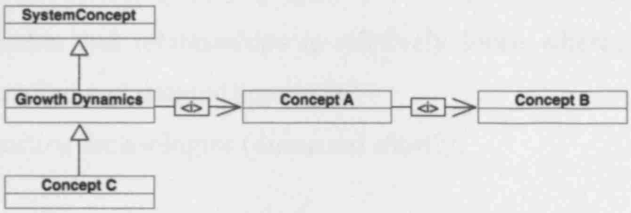
4. *Portability.*
    - *Preferably the knowledge that Φ represents should be portable to alternate applications, e.g. via some shared standard.*
        - Figure 7.6 was constructed simultaneously with a corresponding ontology in Protégé 2000. Though strictly speaking this is not an ontology-building exercise[2] due to the fact that this is a well-structured technique that is based on a set-theoretic approach it lends itself to a similarly structured language, e.g. an XML derivative.

[2] Note that the aim is not to explicitly define what behaviours and concepts are – only to relate them with each other using generic properties. Concepts by themselves only attain a meaning once related to other concepts. Moreover there is a clash between the meaning of subsumption in ontologies and the meaning of the <*includes*> property – OWL and related languages are therefore not a good choice for portability.

It can be argued that point 2 above introduces ambiguity into this technique in the sense that association of concepts, i.e. expert knowledge, is subjective to the user who is interpreting the semantics. This ambiguity should not manifest at the individual model stage, i.e. developing a BIT for a single model – the BIT should only contain the concepts that the original model represents (whether or not it is congruent with the user's belief of the system). This does, however, present itself when integrating BITs together. Take, for example, the generic nature of system concepts in a BIT from hypothetical Model 1:



And subsequent integration with Model 2, which introduces Concept C:



Concept C now implicitly includes Concepts A and B due to the transitivity of the *<includes>* property. It is conceivable that conflict can occur if Concept C is indeed a child of Growth Dynamics, i.e. it fulfils the property of inheritance and is validated as such, but does not include Concepts A and B in reality or the association is explicitly negated in Model 2. Though this may at first seem to be a flaw, it does actually highlight one of strengths of the KDA technique with respect to model integration – if such a conflict does occur it would strongly imply that the two models that are being integrated are in fact incompatible as far as current knowledge and belief of the system and the concepts in question are concerned. One would certainly not want to integrate models that have any fundamental differences in belief of the way the system works – not only does this cause conflict on the level of KDA, such a conflict would manifest at the ABI

level in the form of conflicting rules or agents (ι). KDA therefore can explicitly highlight conflicts in association of concepts.

8.3.1.3 Novelty, Current Works and Future Developments

The idea of describing model semantics is not entirely new. From the outset there were a handful of supporting technologies including:

- Category theory.
    - Abstract and flexible, this method can describe mathematical relationships and morphisms. However it lacks the ability to relate the meaning of the models to the real system.
- Semantic networks (SNs).
    - Shows real promise in describing the relationships between concepts. Indeed KDA shares many ideas with these, however SN representation of concepts and relationships is relatively loose whereas KDA presents a structured and defined approach.
- OWL-supporting technologies (discussed shortly).

Model description has taken shape in as many varied forms as the fields in which they appear. Indeed different fields have had different requirements and hence diverse forms of model representation, as explained in Chapter 4. Almost all have taken a software approach, e.g. MSI and simulation communication technologies from DARPA[3].

KDA in Systems Biology

In the field of systems biology there are at least four standards for model description that are predominant or emerging:

- **SBML** (Hucka *et al.*, 2003).
- **BioPAX** (Stromback & Lambix, 2005).
- **CellML** (Lloyd *et al.*, 2004).

[3] Defence Advanced Research Projects Agency (http://www.darpa.mil/)

- **CMDL** (Margoninski *et al.*, 2006).

SBML was originally meant for general model representation[4] but evolved into the exchange language of choice for molecular pathways. The XML-based language separates two useful representations: pathway components and their mathematical relationships. SBML is however limited in its mathematical representation by MathML, as is CellML, though this does not yet pose a problem as most models in systems biology are P/ODE based for which MathML copes well. Importantly, the planned Level 3 version of SBML will enable submodels to be embedded within larger contexts. This is aligned with the idea of model integration and indeed will encourage integration of sub-models (modules). However, note that such integrations will always be linear – the formalisations in §5.3 wholly describe this. Additionally there are no plans to include any description of model semantics though it can be argued that some of these are implicit, e.g. compartmentalisation of reactions and modularisation of (sub)models. Conversely, KDA cannot describe mathematical relationships explicitly but can describe model semantics and hence these two techniques highlight two very different angles from which model integration can be approached:

1. The traditional approach, wherein the mathematical formalisms are dealt with directly. The possibility of model integration is severely limited in this case by paradigm, focus and scale differences as described in §4.3.
2. The conceptual approach, wherein the mathematical relationships are considered separately with respect to focus.

Like KDA, BioPAX attempts to connect models to focus. Widely promoted as the next step in describing pathway models after SBML its complexity, in the opinion of the author, has hampered uptake by the systems biology community. BioPAX has gone a step further than SBML in that not only does it allow mathematical relationships to be described, i.e. pathway models, it simultaneously attempts to build an ontology of

[4] Andrew Finney, personal correspondence.

components. This is in line with the idea of having a computable[5] form of focus as KDA does. However the responsibilities of mathematical *model description* and *model integration* have been delegated separately to KDA and ABI respectively in the work presented here whilst the authors of BioPAX have attempted to do this concurrently (though admittedly model integration is not the primary aim of the BioPAX project). Moreover KDA does not enforce an ontology as such on the modeller whereas BioPAX uses a fixed set of concepts agreed upon by a number of experts[6] and therefore enables individualised model integration exercises. However, this highlights a disadvantage of KDA in that once a number of iterative model integrations have been performed the corresponding BIT diagrams and concepts therein may well vary when compared to other modellers' representations – the advantage of disambiguity that BioPAX enforces is lost. On the other hand, there is no reason why future developments of KDA should not adopt a strict approach so that `SystemConcepts`, `Systems` and `Components` in particular are standardised. Indeed, upon closer inspection, the vast majority of concepts in BioPAX already fall neatly under the definition of `Component` and so a future integration of these projects is highly desirable. Such an integration will allow modellers to use standardised components that can be translated into Λ and reduce the chances of conflict between modelling groups.

Similar is the case for CellML, which can be broken down further into two languages, AnatML and FieldML, which describe anatomy and mathematical properties respectively (though it seems development in both of these has slowed of late). Indeed AnatML seems to be a macroscopic version of what BioPAX aims to do on the molecular level, i.e. description of what is defined as `Component` in KDA, whilst FieldML was originally designed to express fields such as finite elements or splines (and hence is a good platform on which O/PDE solutions can be described). Again, the explicit mathematical components that CellML can describe are missing in KDA and conversely the expressive

---

[5] Inference engines can be used to draw relationships between components and hence mathematical relationships can be exposed.

[6] In fact cancer systems biology is taking a leading role in ontology-building and disambiguation of biological knowledge. The drive for therapeutic improvement has yielded a number of projects such as those initiated by the NCI (discussed shortly) and the CViT project for example (§3.4.2).

power of system concepts is missing in CellML (barring the expression of ι, which AnatML specifically aims to state).

Whilst SBML, BioPAX, CellML and KDA have all focused on the model and system the development of model integration technology by members of the Beacon Project[7] at UCL currently aims to conceptualise the wealth of data that is used, e.g. experiments that yield parameterisation values for models, in the form of a well-structured repository (Saffrey *et al.*, submitted). This is a very important point – it provides the means to explicate where data comes from and how it was derived, and incorporating this with the idea of a lightweight model execution environment (Margoninski *et al.*, 2006) affords the possibility of a full MMS/IME as defined in §4.4.1. The supporting language, CMDL, conforms to Formalisations 5.1 and 5.3 and therefore represents a linear integration strategy[8], albeit a more complex one than that of SM and other previous works – this is because CMDL provides multiple interfaces for each submodel, i.e. the solver is separated from the actual model. Note that model interfaces and connections need to be provided, as expected in any linear integration. Though engine wrappers and connectors can be made generic enough to deal with individual models on a paradigm-by-paradigm basis[9] it seems that the development of this project will be contingent on building a rich enough set of wrappers to deal with the huge diversity of model types in systems biology. In contrast, KDA is independent of paradigm since the point of using two steps (KDA/ABI) is to separate the model meaning and model implementation.

Interestingly CMDL's concept of a biological phenomenon ("aspect" in the original metamodel, see Figure 2 in Finkelstein *et al.*, 2004) appears to be concordant with `SystemConcept` in KDA although the `phenomenon` element in CMDL contains textual information – again, this highlights the field's current stance to that which is described as focus and scope in this thesis. Much of this metadata is kept in textual form and precedence has been given to the mathematical properties of the model itself. In

[7] http://grid.ucl.ac.uk/biobeacon/
[8] "Contexts" (inputs) are fed into the "engine" (solver), which provides an "interpretation(s)" (outputs), which can in turn provide context for another submodel(s).
[9] As far as author is aware, only P/ODE have been tackled so far.

contrast by defining simple, extendable components KDA provides a set-theoretic way to describe aspects and hence make these computationally meaningful[10]. Additionally the `compartment` element in CMDL bears close resemblance to the `component` and *<partOf>* relationship in KDA. There is therefore a possible point of bridging between these methodologies here. In the opinion of the author CMDL can greatly benefit from a straightforward implementation of KDA in addition to the aforementioned textual information in the `phenomenon` and `compartment` elements.

CMDL/KDA as a whole could therefore provide model description, and thus a standard of model exchange as well as a platform on which ABI can be automated (discussed shortly), at a level that is not yet present in the field of systems biology.

This does however raise the question of controlled vocabularies, i.e. consortium-driven agreement on `SystemsConcepts`, as was the case with BioPAX. The rapid expansion of controlled vocabularies such as the internationally driven open-source EVS[11] project from the National Cancer Institute (NCI) is a good start. Though this does not address model foci it does hold promise in terms of internationally coordinated efforts to "ontologise" cancer-related data. In fact the National Cancer Research Institute's (NCRI[12]) Informatics Initiative aims to create coherent and compatible informatics platforms enabling the deposition of a wide range of accessible cancer-related data – this is certainly an important step forward in bringing expert medical knowledge to systems biology, and hopefully in explicating a diverse set of `SystemConcepts` for use in KDA.

One of the greatest disadvantages of the KDA methodology is the fact that every model, which is usually published either as plain text in a paper or put into a modelling language such as SBML, must be re-conceptualised into `SystemConcepts`, etc., manually. There is no possibility of automating this since none of the exchange formats provide a way to describe these concepts (and hence there is no obvious mapping) and of course it

[10] KDA would obviously need to be implemented into a language, e.g. based on XML.
[11] http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/vocabulary
[12] http://www.ncri.org.uk/ and http://www.cancerinformatics.org.uk/

is challenging to extract this kind of information from plain text (even assuming all the information that is needed is there). However, the benefits of manual generation far outweigh the effort. Along with the mathematical description the connection a model has with reality is concretised into a format that software can then begin to use intelligently. In the opinion of the author this is one of the next great challenges for systems biology – not only to model biological complexity but to provide intelligent software systems that can infer knowledge about the biological system, extract and present information and hence provide the means to understand the system. Ultimately in the case of cancer research this means determination of therapeutic advantage by (i) enabling model integration and hence a complete simulation of tumour behaviour and (ii) explicit definition of system concepts and their relationships leading to better human understanding. An example of why such intelligent software systems are needed has been alluded to earlier in this section where the transitivity of the *<includes>* yields relationships between `SystemConcepts` that were not explicitly stated in the original models. As more and more models are expressed in KDA the associations between elements become massive (as seen in Figure 7.6) and hence a software aid becomes necessary. Indeed such an inference technique would represent a truly exciting future development for this project – as $M_I \rightarrow \Phi_I$ one comes a step closer to possessing a computational model of the system, not in the mathematical sense but in a way that enhances the ABI strategy (discussed shortly). With that in mind considerable effort was put into searching for an appropriate technology to demonstrate this. Two immediate candidates were Protégé's inbuilt inference engine (which was subsequently dropped because of irregularities), KAON2[13] and Jena[14], which are both open-source projects. There are other packages but most are commercial or lack support.

Overall KDA has successfully fulfilled all design requirements, enabling both model (and therefore system) description and ABI; but what is especially exciting is its future applications. The integration of ideas across CellML, CMDL and BioPAX would yield an

[13] http://kaon2.semanticweb.org/
[14] http://jena.sourceforge.net/inference/

extremely rich language that not only provides the ability to exchange models but also enables model reuse, integration and redevelopment.

8.3.2 Agent-Based Integration

8.3.2.1 Model Decomposition: Genetic Algorithm Performance

At the core of the ABI method is the optimisation algorithm that can enrich an initial set of rules to perform in Λ. The reasons for choosing GA were as follows:

1. A large solution space was always expected, even when starting off from a good initialisation set. This is the trademark problem for which GAs in particular have proved to be especially good (Mitchell, 1996a, Chapter 1).
2. The structured and compartmentalised form of string-based rules made GA a natural choice when considering how to perform mutations, crossover and mating on rule sets, e.g. mutation on numbers and operators and hetero/homogenous crossover between rules.
3. There is a history, albeit a brief one, of use of GA in rule optimisation for CA (discussed shortly) and hence it was already a proven technology for such use.
4. The evolution of rules can be tracked, i.e. logging the evolution of each individual at each generation which could indicate how best to perform optimisations of a similar kind in future.
    a. A corollary to this is that GA is not a black-boxed approach.
    b. This is not yet implemented in the software produced here but represents a very important future enhancement to the software.
5. A number of candidate solutions can theoretically be found.
    a. Whether this is an advantage or disadvantage depends on how the results are used. If candidate solutions turn out to be very similar (in terms of structure and magnitude of variables) then there is no problem. However if there are hugely varied rule sets then it must be up to the modeller to decide what is biologically feasible based on current knowledge.

6. GAs are very configurable – the GA described in Chapter 6 had 39 configurable attributes based on probabilities for mutation, crossover, mating (and sub-attributes thereof consisting of operator types) as well as choices based on fitness function attributes.
    a. Whilst this can be a great advantage the reverse argument is that too many choices can lead to a situation where the optimal configuration for a specific experiment is difficult to determine. This is the reason why a benchmark was necessary, i.e. to determine the best conditions for rule enrichment. It must be noted, however, that though the same configuration is not expected to be optimal it does at least give an approximation. In other words, the configuration for enrichment for the diffusion model rules is unlikely to be the optimal conditions for other more complex models but at the very least indicate an interval in which one can avoid GA individual extinction or a hanging population.
7. Fault tolerance – a small number of good individuals amongst a population of bad performers can eventually come through after a sufficient number of generations. This is levied against the stochastic characteristics of the approach in that good performers may well loose their advantage due to random mutation.
8. Relatively straightforward to program and integrate with the simulation software.

As with any method there are disadvantages:

1. There are potentially a massive number of simulations that must be performed to find the optimal solution – especially since the starting population in most of the experiments performed here consisted of 100 individuals. This can make single run times arbitrarily long. This was combated in three ways:
    a. Simultaneous application of the fitness function. The original model was run alongside the simulation and the simulation logs were sampled periodically to check for key states, e.g. population number. A cache of best performers at the same time points were stored. If the current simulation's state values at those time points had an error larger than that

of worst performer in the predefined subset of best performers then it was immediately rejected and simulation was stopped. Otherwise it is added to the best performer list, and displaces the bottom-most individual from that list.

   b. A cache of the string-based rules so far performed is stored along with the performance of that set. If the same rule set was encountered the simulation was not performed.
   c. Parallelisation: 15 machines were used and split into 3 quintuplets to perform 3 simultaneous GA instances.
2. Too many configurable components (see earlier point).
3. The model runner embodies the traditional problem in model integration, i.e. the need for a communicative interface.
   a. This would be the case for any method.
   b. Integration with SOAs would greatly reduce this problem (discussed in §4.4.2).
4. Use of GA may well be excessive for some decompositions.
   a. GAs are especially good at finding solutions when there are many variables.
   b. Other methods can be used since those parts of the software were designed generically.

Optimisation is an entire science unto itself. There is a plethora of methods that are continuing to be developed. GA was chosen as the most natural and straightforward choice, however there are several alternatives that could be used considering the qualitative as well as quantitative optimisation that is sought:

1. Neural Networks, Bayesian Networks and other learning algorithms.
   a. There is a vast array of neural network and other learning methodologies catering for a multitude of problem sets. However it was not clear how a learning algorithm could have been implemented considering the core

elements involved, i.e. an initial rule set, a target behaviour and a required target rule set.

b. One way in which these could yield fruitful results is the learning of relationships between rules that have already been optimised and original model attributes, e.g. parameters. This is desirable because the ABI strategy yields a result (rules) for a particular model instantiation, not a model class (as defined in §4.3.1). The more generic rules can be the more powerful ABI is as an integration tool.
c. Another way these can be applied is reinforced learning of rules in agents during simulations. A disadvantage of this method would be that rules would be black-boxed. An outline of the workflow would be:
    i. Initialise the agents in Λ with naive nets embedded in rules, i.e. inputs would be states/environment, etc. and output would be next state/location for agent.
    ii. Begin simulation.
    iii. Each time step is evaluated by fitness function.
    iv. Good/bad performance back-propagated to nets.
d. Effort was put into implementing a GA-modifiable module for this project by adapting Joone[15], though it was never put to use.

2. Artificial immune systems show some promise in searching large state spaces. The principles that govern the immune systems learning are, however, quite similar to that of evolutionary techniques. They have been especially identified as good candidates for nonlinear pattern recognition and hence may hold more promise in recognising behaviour concordance in the fitness function rather than an optimisation technique itself (de Castro *et al.*, 2002), as well as in a similar role as that described above in point 1b.

Decomposition of the diffusion model as a benchmark proved to be worthwhile. Relatively few GA operator configurations, 12.3% to be precise, could actually yield a positive result (though this only accounted for 3 configurable components – see

[15] http://www.jooneworld.com/

Appendix C1). As expected there had to be a balance between operators to avoid an extinction or hanging population. Admittedly more needs to be done to determine the optimal conditions for best results for a given model complexity – all 39 configurable components need to be investigated more thoroughly.

Other Improvements

An alternate line of investigation that should be undertaken is in the way GA actually integrates itself with the simulation environment. A vast improvement was observed simply by reprogramming the fitness function to simultaneously measure fitness during the simulation process.

The idea of more efficient and smaller simulations could be taken further by borrowing from Patch Dynamics theory (§4.3.1.3). For example, let Λ be the simulation space with I interactors and $\iota_i \in I$. Also let space and time in Λ be discretised, $\partial_{xyz} \in D$ and $\tau_k \in T$ with both ∂_{xyz} and τ_k considered intervals. The original model can be abstractly symbolised as $f(d,t,\boldsymbol{X})$ where $\boldsymbol{X}$ is a vector of inputs (parameters or inner functions). One can now develop a new GA scheme wherein fitness is evaluated at predefined patches in the simulation, both in space and time, such that $f(\partial_{xyz},\tau_k,X_{xyz,k})$ is evaluated for its micro-behaviour in Λ but not in intervals in between. This would greatly reduce simulation and evaluation time as a side effect but more importantly this raises an exciting avenue of investigation. Decomposition of behaviour of a model in its entirety could hinder the progress of a GA; the burden of this complexity is lifted from the GA in this new approach and could lead to more positive results for the most extensive and complicated models.

8.3.2.2 Case Studies and Analysis of the KDA/ABI Strategy

Case Studies

The diffusion model was chosen as the first decomposition since:

1. It is a relatively simple PDE.

2. Many models incorporate it, or variations of it, and hence it is a valuable starting point.
3. It has a strong theoretical basis.

The movement of molecules, metamodelled by lattice state changes, drove the need to create specialised `State`-implementing classes that reciprocated molecular movement. This was done in response to initial experimentation where the GA could not develop the right combination of simple rules that would ensure mutual, i.e. equal and opposite, state changes in adjacent sites, hence conserving mass. In other words if there is c flow from lattice l_1 and l_2, where the rule being executed belongs to l_2, there must be a corresponding loss of c at l_1. This was an interesting result in itself in that such an optimisation would have required a co-evolutionary strategy, i.e. a rule for incoming and a rule for corresponding negative flow between the same lattice sites. This is a clear indication of the criticality of three points of the optimisation process:

1. The importance of an optimisation algorithm that can co-evolve such rules.
2. Prior knowledge is needed and embedded in initialisation of rules – in this case, conservation of mass.
3. The dependence on software components that can aid the optimisation process – in this case the `State`-implementing classes that automatically ensure conservation of mass.

The GA traversed more than 1700 simulations[16] before reaching a solution. It is interesting to see the basic form of the resultant rules (translated into pseudocode from the tabular format described in §6.2.3):

- If{true} move in x,y,z-direction $[((D/d) \times c)/a/b]$

Where a and b are constants, D/d is the given diffusion constant as a ratio of d and c is the concentration in the parental lattice site. In actuality all results were more complex

[16] An exact number cannot be given since logging was intermittently suspended for efficiency.

than this but could be distilled to this form. For example, due to chromosome replication and mutation, some rule sets were found in duplicate but their mathematical effect could be distilled to the form shown above[17]. The best results for a were found to be 5.698, 5.744 and 5.898; best results for b were 1.976, 2.132 and 2.134 (all values given to 3dp, actual values in 14dp). This is a very exciting result as it closely reflects the theory on which Fick's 2nd Law is actually based. D/d represents a resistance of molecular flux based on the interface between lattice sites – a von Neumann neighbourhood was employed, hence the divisor that tends to 6 (a), i.e. the number of neighbours through which there is a 2D area that flux can pass, and correspondingly there is an equal chance of flux in the negative direction for each pair of lattice sites based on the current concentration ($c/2$) and hence b tends to 2 (Nelson, 2003, Chapter 4).

The significance of this result, even though this is a relatively simple model, cannot be overstated. Not only has decomposition led to a rule set that solves the original model the rule set actually reflects what is happening on the molecular scale[18]. Yet this result should not be surprising – with the right initialisation rules and conservation rules implicit in states it should be natural that such a simple local interaction be evolved (even though the actual forms of the rules in the successful result sets were more complex).

The Gompertz model, being a phenomenological model, could only be decomposed by KDA into a very small set of concepts. Moreover the model does not yield any information about the mechanics of the system and hence there could never be any determination of ι (barring population number and hence cells as agents) as per the process described in §5.3.4. Such being the case there was little for the GA to operate on and establish a successful rule. Indeed the initial rules that were used to find a solution were based on the fact that the curve is rate dependant (i.e. it can be expressed as an

[17] An inbuilt mechanism to reduce rule complexity would be an excellent addition to the GA software. Not only would this simplify rule interpretation by the modeller it would greatly reduce execution time since the number of rules to be executed is potentially cut (certainly in the diffusion case the rules were repetitive, often identical, with mathematical components that could easily have been concatenated by simple arithmetic.

[18] Note that the best results have tended to the theoretical values of 6 and 2 for a and b respectively but not reached exactly. This is probably a side-effect of using the Euler difference method, which accumulates error with each progressive step.

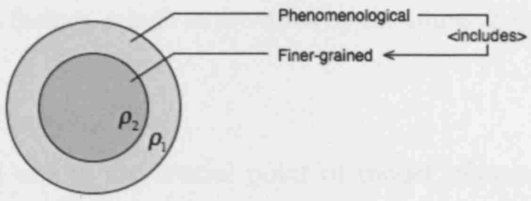
ODE). It is interesting that none of the 18 GA runs yielded a result. In a separate 2 experiments the population of cells was implemented as a global variable with the same rate-dependant rules, i.e. to simulate Equation 2.1. It would seem logical to assume that the GA would be able to find a solution to a relatively simple model however even in the 2 runs with rate rules on the global variables a solution could not be found. This could be a case of a GA being an excessively forceful method seeing as there are relatively few variables to operate on. This is reminiscent of the findings of Bossomaier *et al.* (1999) where simple binary CA rules could not be optimised.

A successful result could only be procured once finer grained concepts were introduced. This highlights an aspect of the KDA/ABI strategy that has already been mentioned earlier, which is that the method does require human interaction and knowledge to complete integrations. This is not however a major disadvantage – human interaction is always expected. Even with the linear integrations described by Geoffrion (1989), Dolk (2000), Villa (2001) and Takahashi *et al.* (2004) a certain amount of human input has always been needed. In conclusion it has to be said that inputting additional knowledge into the BIT that is not present in the original model is not in KDA's mandate. This is perhaps a limitation of the approach in that purely phenomenological models do not yield enough information.

All subsequent experiments were extremely encouraging. As explained in the previous chapter, in the proof-of-concept implementation, the Ribba *et al.* model was simplified for ease of implementation, though note that this would not have affected the integration process aside from computational tractability. The pathway model was left out specifically because of the large number of components (i.e. molecular species) that were involved, which would have had to be implemented to each cell (agent) individually – obviously with the limited resources available (in fact even if there were more nodes) this would have imposed limitations on the integration. This is probably why the original model was implemented in 2D, since the number of possible cells in the simulation space given the population-based growth constraint would have only been a fraction of the number observed in a 3D space. Nevertheless it must be said the pathway model,

implemented as a Boolean network, could still have been implemented with a metamodelling approach, such as asserting same-state mechanics within lattice sites. This is exactly what was done with population number with this model, i.e. instead of modelling cells individually population number (separated by cell cycle stage) was implemented as a lattice state. As the results show this actually works very well, especially since the simulation successfully imitates the behaviour of the original model, and is in fact a viable solution to the scale problem.

Integration between the Ribba *et al.* model and the Gompertz model was relatively straightforward seeing as the Gompertz `SystemConcepts`, being entirely phenomenological, subsumed the behaviours of the Ribba *et al.* `SystemConcepts`. Under the belief framework it must be assumed that phenomenological-derived rules and parameterisations must take precedence over child rules and parameterisations. This is because in KDA the use of the <*includes*> property asserts total behavioural inclusion. This idea is best illustrated graphically:



KDA asserts the rule set relation $\rho_1 > \rho_2$, i.e. there exists an unknown rule set of unknown concepts that additionally, along with ρ_2, contribute to the behaviour elicited by ρ_1; ρ_2 can never fully explain phenomenological behaviour and thus in a belief framework the parameterisation and rules of ρ_2 must be superseded by ρ_1. Hence in the case of Gompertz/Ribba integration preference of population thresholds and cell cycle lengths was given to the Gompertz rules.

As Figure 7.17 shows the integration has had the effect of shifting the original Ribba curve, which was less constrained by population, towards the Gompertz curve (although admittedly the original Ribba *et al.* model was almost Gompertz-like already). The tighter

population constraint also explains why the maximum growth is achieved earlier and lower. Also, the introduction of a death term ensures that cells that fall into G_0 do not simply revive and continue growth. The additional complexity of rules contributed by the Ribba model maintains its original behaviour, which is a significant finding – had the system been chaotic in the sense described in §3.3.1 one would expect erratic fluctuations due to changes in initialisation or rules. Instead, as the original Ribba *et al.* model showed when vessel number and other characteristics were changed, the behaviour is not erratic. This is a key property that separates "random" and chaotic systems from ordered and structured systems.

Note that though the Gompertz model is wholly phenomenological, and that one should not expect its behaviour to contribute any value to a finer-grained model that is totally subsumed by it, the integration between the Gompertz curve and the Ribba *et al.* model has yielded information about the behaviour of *both* – the dependence on population constraints and cell cycle length in the Ribba *et al.* model is highlighted and it has now been shown that a more Gompertzian-like growth can be achieved by finding the right balance between these factors, which is provided by the integration.

Evaluation Based on Case Studies

These arguments lead one to the crucial point of model integration – does integration actually yield a model that is of more value? Does it shed any more light on the behaviours of the original models and the behaviour of the actual system? In this case:

- The resulting model yields a richer behaviour than the original parts.
    - Neither model show similar behaviour, aside from growth in population and an eventual levelling in the Ribba *et al.* model.
- The Gompertz model does not take into account such things as oxygen levels, but are taken into account macroscopically as population constraint. This can now be divided into both population constraint by neighbouring numbers and constraint by hypoxia. In effect the Gompertzian explanation of the microsystem is richer.

- The Ribba *et al.* model shows unrestrained growth. Integration has yielded a model that is far more familiar to MTS growth since this is what the Gompertz model was validated against.
- The resulting simulation is now defined on the local level, which is more than what the Gompertz model could describe.
    - Variations on parameterisation, such as vessel number, oxygen diffusion, etc., can now be simulated with relative ease. These only require a change in the corresponding parameter in rules and the initial setup, e.g. see Figure 8.1 below.

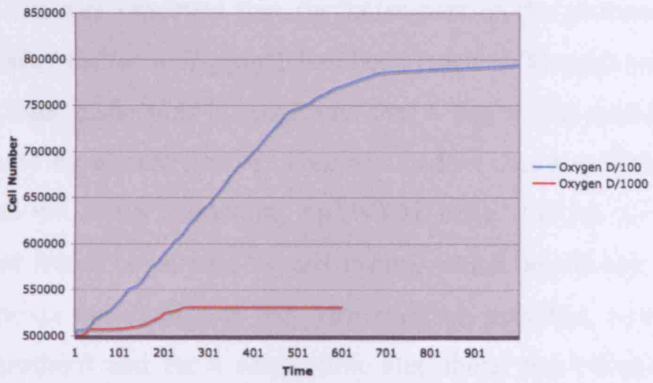


Figure 8.1. The effect of decreasing oxygen diffusion coefficient (compare to Figure 7.14). Population number is drastically reduced (the same setup with 1 vessel was used as described in §7.5.1.5). The growth shape is maintained since the result of reducing the coefficient by a factor of 100 effectively reduces the capacity of oxygen to reach its targets in a measure that is related to the coefficient and hence the hypoxia threshold is reached far faster (blue line). A reduction of a factor of 1000 cripples growth almost completely (red line).

The integration between Ribba *et al.* and the Gompertz function was a proof-of-concept, however one can imagine a scenario wherein the Gompertz or logistic curve has been fitted from actual data, which then needs to be integrated with a pre-generated model – in fact this data would be a representation of the real system and in every sense fulfils Definition 4.1. Curve fitting is a regular and necessary step, as explained in §2.2.1.3, where a model will be developed first and parameterised after experimental results have become available, e.g. Chignola *et al.* (2000) attempted to parameterise their growth model with MTS data so that it would behave sigmoidally. Viewing data as a model of the system, this can also be viewed as a model integration of a kind. Chignola *et al.*'s

model was relatively simple and only involved the optimisation of two parameters to explain growth, i.e. proliferation and death. In the Ribba *et al.* model case the pre-defined model was a mature description of the system already and the ABI method was able to enhance its behaviour, i.e. fit it to a known growth behaviour. It is quite conceivable that instead of using a model at all in this methodology, one also can integrate (i.e. optimise rules) directly from experimental data. Whilst this is possible theoretically it has not yet been attempted here but is certainly a line of future study. This highlights the versatility of the ABI method.

The Ribba/Gompertz integration also draws attention to peculiarities of using this discrete method – it was observed that the latter part of the procured curve actually fluctuated in rate after the limit of growth had been reached. Though growth had seemed to steady, upon closer inspection it was found that a lattice site could often exceed its population threshold by almost double. This was in fact because of the metamodelling methodology used to avoid modelling individual cells. Lattice site states denoted population for that lattice (separated by cell cycle), which would halt growth once the population or hypoxia threshold was met. However, on occasion, when the population was below the threshold and for a single time step there was enough oxygen for the hypoxia state to be set to false a small fraction of cells that had migrated from neighbouring sites had the opportunity to multiply despite the population threshold (since that logic step had already passed). If one were to assume that the population threshold was purely a space constraint then of course such an occurrence could never physically take place. However note that the explication of KDA is advantageous in this situation. Indeed these very caveats highlight the need to integrate knowledge together – in this case additional rules need to be set, or a reordering of logic is necessary, that redefine proliferation and cell movement such that space constraint is not violated. This method therefore not only enables model integration but also encourages further integration by highlighting missing behaviours[19].

[19] Of course this is always down to interpretation and flexibility – it might be the case that the inclusion of additional behaviours is not necessary, depending on the application.

Conversely it is also possible to exclude behaviours deliberately. In this example the pathway module was left out as well as the radiotherapy module. Note that both can still be included if desired: the pathway model was discussed earlier; the radiotherapy model is governed stochastically with cell death or cell recovery occurring according to two probability distributions (simulating single and double strand hits) based on the pathway model status. This could have been implemented as follows within the existing rules:

- If {not in G_0} Update pathway model
    - See Boolean updates, Table 2 in original paper.
- If {true} generate random number.
    - If {within limit [0,x]} single strand break.
    - If {within limit [x,y]} double strand break.
    - If {within limit [y,1]} no break.

This particular part of the model can now be excluded by simply removing the rules that were derived for the module. Modularised models are relatively straightforward to dissect in this manner. Of course this is also feasible in linear integration, if inputs/outputs can be matched up, however the difference is that now individual rules (i.e. behaviours for particular agents) can be dissected with more ease and within the context of the real system.

The successful integration of the angiogenesis model was an encouraging development since many models in the literature have fallen short of describing vascularisation. In this particular case the stochastic solution, i.e. a biased random walk, devised by Anderson (2003) was completely adaptable to the simulation software and no GA optimisation was needed. It should be noted that a larger simulation space is needed as this was the only limiting factor to growth. However the integration itself was seamless since all rules represented RC4.

The growth curve is reminiscent of other vascular models in the literature (see Matzaris *et al.*, 2004) where growth is less-than-smooth, unlike the Gompertz curve, but nonetheless

exponential. The exponential phase was uneven, as seen before, but this was now additionally due to erratic oxygen supply. After that phase, however, models tend to differ depending on the parameterisation used – for example, Zhang *et al.* (2007) report a termination of growth as constraining factors limit gliomas, as seen here, whereas Arakelyan *et al.* (2002) report unrestrained growth. This highlights the fact that different models have been validated by different data and this must be taken into account during the integration process – indeed KDA should reflect this information.

Correspondingly there is a general trend of growth in the number of hypoxic cells in the simulation space. The oscillations seem to be occurring with the course-grained rule updates of vascularisation (every 36h) causing spurts of expansion. The point is that the integrated model now shows vascularisation as well as growth, which would be something not possible to do without a lot of effort (or starting afresh) from the original models. Certainly the resultant model is producing interesting, biologically relevant behaviour – the hypoxia curve in particular is of interest. It is a well-known fact that hypoxic cells are particularly resistant to radiotherapy (Stamatakos *et al.*, 1998) since they are arrested in the cell cycle and therefore determination of the distribution of both hypoxia and cells by cycle position in a vascularised tumour is an extremely useful metric. This integration affords the possibility to achieve this since the original models separately contribute oxygen distribution, cell growth, cell cycle and vascularisation.

As always, there are improvements that can be made relating to both the integrated model and the simulation software itself:

- TAF production needs to be introduced.
    - The vessels grew up the TAF gradient, however once TAF had depleted EC movement was more random.
- Fibronectin production needs to be introduced.
- Blood flow
    - As explained in §2.2.2 blood flow itself can shape vascularisation and have a dramatic effect on overall topology. A number of LB models, as

well as PDE models, of this kind exist and so it is well worth investigating integration of these models.

- Vessel maturity
    - Arakelyan *et al.* (2002) point out the importance of vessel maturity and its effect on oxygen distribution and tumour dynamics. Currently vessels are assumed to be immediately viable, which is not correct in the biological sense.
- A finer-grained lattice is needed
    - A vessel is currently defined by a Boolean state for that lattice site (i.e. the entire lattice site is converted to a vessel). A finer grained approach could be to have more inner lattice sites, however introduction of shaped agents that can traverse the simulation space would be a better option.
- 3D Visualisation and analysis components
    - This would greatly aid the analytical power of the ABI approach. Normoxic and hypoxic distributions *in situ* with vascular distribution would mark a significant step forward when coupled with the fact that the integration process can add even more behaviours.

8.3.2.3 KDA and ABI as a Whole: Evaluation and Future Directions

The core aims of the envisioned integration strategy for this project were:

- To perform integrations that are model paradigm independent.
    - This is ensured by the decomposition into local interaction rules.
- Non-violation of focus and scope.
    - This is ensured by KDA.
- Simultaneous means to a 3D visualisation.
    - Though not yet implemented this is possible to achieve.

Analysis and Comparison With Other Methods

An optimisation method that translates the model into local interaction rules ensures paradigm independence. However one could argue that the job of paradigm integration

has been offset to the optimisation algorithm. To some extent this is true – for all integrations described in §7.5 the `FitnessFunctionModel` implementation object worked in conjunction with a specialised `ModelRunner` implementation with which it was comparing results. Of course writing a specialised implementation for each and every model reduces to the same problem faced by any SOA-like integration strategy such as SBW, i.e. interfaces for every model.

Indeed this very track was explored early in the project with the development of SCIpath (Patel & Nagl, 2004), formerly MicroCore, which was originally engineered for pathway visualisation and capability for eventual model execution. The need for an integration strategy was quickly identified and a SOA-like architecture was therefore devised under which different types of data could be exchanged in a structured way. The implementation involved exchange of basic data types (`char`, `float`, `boolean` etc.) in lists or matrices, communicative objects such as ODBC drivers, SOAP interfaces, model language interfaces such as SBML (exchanging nodes and reactions) and a GUI API (for remote GUI invocation). Due to time constraints this could not be developed further, however with SBW and the MMS-like system devised by Saffrey *et al.* (submitted) it is now possible to envision a process chain, as illustrated in Figure 8.2, wherein application-level integration tools use their own solvers, or the model's solver, to run/simulate the model and the rule optimisation engine operates under the SOA by querying for data. Such a coupling of technologies would mark a state-of-the-art in model integration for systems biology and represents a real solution to a very complex problem.

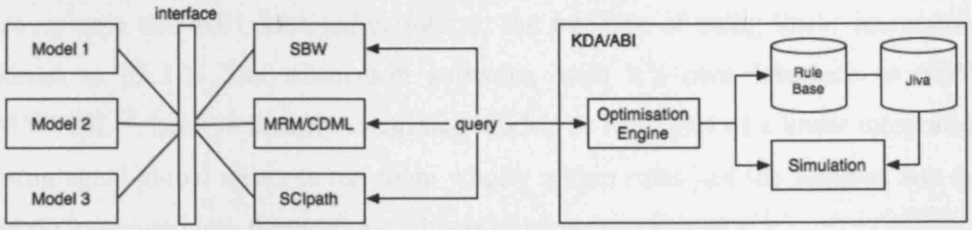


Figure 8.2. Desired integration between SOAs and the software produced in this project. SOAs communicate via interfaces to models and are able to extract data based on a query (implemented as, for example, SQL). This provides the information needed to optimise rules, which are eventually stored with associated agents (Jiva).

The questions that are arise are: (i) What does ABI offer that these other software (linear) integrations do not already provide by themselves? (ii) What does the coupling of technologies described in Figure 8.2 provide to modellers that has not already been addressed? Concisely:

- The possibility of model integration when there are few or no connecting variables.
    - Linear integration is limited to those models that can be connected by common variables, i.e. matching inputs to outputs and vice versa.
    - Note that this does not negate the value of software infrastructures such as SBW and SCIpath – the schema illustrated in Figure 8.2 identifies single model interfaces for the optimisation algorithm.
- Understanding on the level at which the real system is actually operating, i.e. local interactivity and agent-level understanding.
    - Provides a more in-depth understanding on the system.
- Simultaneously relates output to agents.
    - Experimentation can then take place outside the context of the original models, e.g. rules for a certain agent can be modified and the effects observed. This was demonstrated in Chapter 7 when varying cell cycles lengths and synchronisation was directly simulated.
- Readily parallelisable (Figure 6.4).
- Possibility of 3D visualisation output.

Moreover note that ABI does not counteract the freedom of using linear integration, as explained in §5.3.1. The simulation software, with it's own interface to SBW or MRM/CMDL[20], can run linearly integrated models or form part of a linear integration by exposing agent/global states or run them wholly within rules just the same as any model would do integrated into these SOAs. This is illustrated in Figure 8.3.

[20] Model Run Manager that uses CMDL (a name has not yet been given to the architecture described by Saffrey *et al.*)

The most obvious advantage, and indeed the driving factor that led to ABI development, is the fact that many integrations cannot be achieved by the linear strategy even when paradigm, focus, scale and scope are compatible. There are two predominant views of this type of integration:

1. The mathematical view wherein integration takes place focusing on the mathematical properties of the models.
2. The software oriented view, which takes into account point 1 and additionally provides the infrastructure of communication of software components.

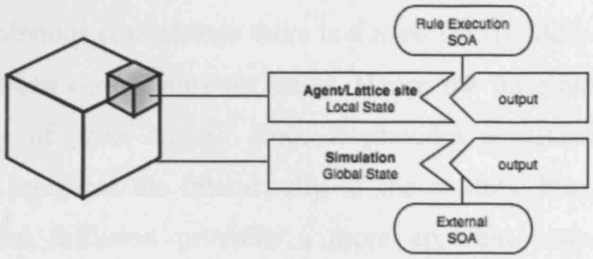


Figure 8.3. Linear integration is still achievable in ABI in two ways. Firstly a rule can be embedded with the logic for linear integration. The input would be the local state and output would affect those states. A corollary to this approach would be a parallel linear integration that influences the behaviour of agents in the simulation. Secondly the global state of the simulation can be linked to linear logic to affect both global and defined local states. In both cases note that integrating SOA-like architectures, with MPI or MSI communication, can achieve a chain of simulation spaces. Hence a far larger scale simulation can be envisioned wherein each component represents, for example, a single system such as an organ.

As a validation of ABI it is worth testing whether the Gompertz function, diffusion, and various components from the models by Ribba *et al.* and Anderson & Chaplain integrate linearly in either view. In a mathematical view (paradigm profiles given on right):

Nutrient Diffusion $\frac{\partial c}{\partial t} = D\nabla^2 c \Big\} \left(f^m, q^m, c^m, \neg s^m, \neg e^m, 3d^m \right)$

Gompertz $Y = A + Ce^{-e^{-B(t-M)}} \Big\} \left(f^m, q^m, c^m, \neg s^m, \neg e^m, 0d^m \right)$

Boolean Network $g[\mathbf{m}] \} \left(f^m, \neg q^m, \neg c^m, \neg s^m, \neg e^m, 0d^m \right)$

Darcy's law $v = -k\nabla p \} \left(f^m, q^m, c^m, \neg s^m, \neg e^m, 3d^m \right)$

$$\text{Anderson}\quad \left.\begin{aligned} \frac{\partial n}{\partial t} &= D_n \nabla^2 n - \nabla\left(\frac{\chi_0 k_1}{k_1 + c} n \nabla c\right) - \nabla(\rho_0 n \nabla f) \\ \frac{\partial f}{\partial t} &= \omega n - \mu n f \\ \frac{\partial c}{\partial t} &= -\lambda n c \end{aligned}\right\} \left(f^m, q^m, c^m, \neg s^m, \neg e^m, 3d^m\right)$$

The Anderson & Chaplain model already incorporates diffusion of TAF. To integrate diffusion of any other molecule, an extra term would need to be put into the system of equations. For it to have any meaning it would need to be related to at least one component of the model. Here is the first hurdle of an unstructured linear approach – in the absence of any obvious connections there is a need for (i) additional knowledge and (ii) translations between connections (or both). Hence the introduction of a KDA-like approach is already of great benefit since it provides a structured, shareable and meaningful way to represent the interactivity of the models. Integration between the Gompertz model and diffusion provides a more apparent connection, i.e. that of concentration of nutrient and cell number and more atomically, *B* and *C*. Again a translation component would be needed to model the connection between growth and nutrients and it is not immediately obvious how this could be implemented. Note that the modules in the Ribba *et al.* model actually were integrated linearly since there was a straightforward connection between the state of the pathway and the decision for growth based on cell cycle (modelled as an equivalence vector of Booleans in the network) and tissue dynamics, which was in part made possible by the particular solver on a 2D lattice. This model is therefore relatively open to a linear strategy in that any one of the components in the chain (molecular, cellular and tissue models) are replaceable so long as the replacement model has the same inputs and output. However even in this case the other models above are not readily open for integration. Hence in the strict mathematical sense none of these models can actually be integrated in any obvious way.

In a software-oriented view the implementation of the models, as well as associated solvers, need to be communicable. In this case the diffusion and Gompertz models, being relatively straightforward to implement in virtually any programming language, should

not pose a problem. Moreover they both can be translated into a communicative language such as CellML in the form of MathML components. Such is the case for the angiogenesis model too. These are straightforward because CellML, and indeed most communicative languages and architectures that enable linear integration, have been tested and validated against DE-based models (and rightly so, seeing as most of the models in the literature until recently have been DE-based). It is of note that none of these models have actually been translated into CellML, SBML or CMDL as far as the author is aware (except for diffusion, which forms part of many models).

On this analysis one can see that linear integration is more suited to "model-growing" situations. A use case would start at a model or set of related models which can be extended by virtue of existing parameters and variables. For example, one can identify key variables that limit the current set and that can be expanded to include other sets of models, and hence the linear approach aids the growth of the model set. One of the great advantages of this type of integration is that models can be kept generic, i.e. in their class form rather than instantiation form. Recall from §4.3.1 that models can exist in these two forms, e.g. the equations given above are all in class form whereas those used in ABI in Chapter 7 were in instantiation form. This means that the linear structure along with the calling sequence will work with any instantiation of any of the modules, whereas the ABI methodology does not have this benefit. Instead rules are optimised for *model instantiations* and hence are only applicable to that instantiation unless a definitive relation between the rule parameterisation/logic and the original model can be found. In the case of diffusion this was straightforward but for more complex models one can expect to find correspondingly more complex relations. This will perhaps be one of the greater challenges ABI faces but a solution represents a great advance in the field of model integration and therefore future research in this particular area is definitely worthwhile.

KDA/ABI provides an abstraction of models such that the modeller can interact with the models regardless of their original basis and implementation. The closest work to KDA/ABI outside of systems biology is that of Vangheluwe *et al.* (2002) who directly

tackles multi-paradigm model integration by providing an abstraction theory involving the iterative deprecation of components to simple ideals. In many ways this is similar to ABI in that a common ground (Definition 5.1) is sought. The theory subscribes to a discretisation of the original model(s) by thresholding outputs into intervals and ultimately into sets of Boolean states (i.e. binary automata). However not all models can be decomposed this way and there is a huge loss of granularity in such a method. As an alternative both Vangheluwe *et al.* (2001) and Zeigler (1993) suggest a chained approach to multi-paradigm integration in which a defined network of decompositions from known formalisms is developed, as illustrated in Figure 8.4. However such a method also has the pitfalls of loss of granularity as well as being constricted to a network that is woefully short of translation functions (especially with the enormous number of formalisms currently in use that are constantly being redeveloped). ABI on the other hand does not suffer from these setbacks: though also using discretisation, understood to be a necessary step by the modelling community (especially in the case of multi-paradigm modelling in complex systems), the common ground is not found through chains of conversions and loss of granularity; the introduction of knowledge and connection to the system by KDA/ABI ensure maintenance of granularity and behaviour.

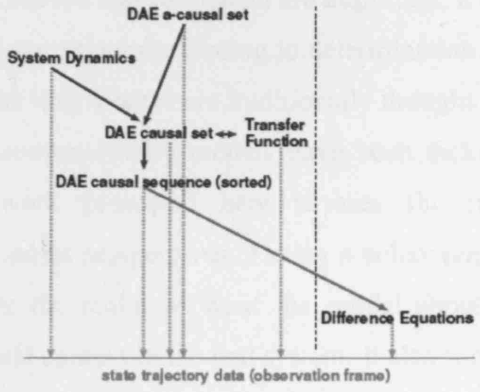


Figure 8.4. Conversion methods are developed for differential algebraic equations (DAE) and other formalisms. This has its roots in the DEVS strategy, which Zeigler and co-workers have continued to develop into large networks of conversions. This suffers limitations in a modelling environment that has grown into many areas of methodologies, as is the case for cancer systems biology. From a practical perspective it would additionally need interfaces between model types. Image taken from Vangeluwe *et al.* (2002).

Impact On Systems Biology

The targeted impacts discussed in §5.2 will not be repeated here but are concisely reviewed in light of results.

Systems biology has been a leader in terms of integrating principles, as seen with the developments of the Physiome Project, SBML, CMDL, CellML, NCI initiatives in ontology-building and informational resources and consortia of experts, most notably CancerGrid[21] and CViT in cancer systems biology, yet a fully integrated *in silico* tumour is still lacking. To the best of the author's knowledge, no universal integrating theory or implementation yet exists in the field of systems biology.

It remains a fact that different methodologies suit different problems and that there are distinct experts for every subfield developing models independently. It therefore seems that both linear and ABI strategies together, along with the required resources, will hold the key to making *in silico* experimentation a reality. ABI by itself, purely on the merit of results described here, represents a great step forward – results have shown that models can be decomposed, though the implementation needs extending and optimising and further tests on more complex models is necessary. Results have additionally shown that, although model implementation and formalism are important, it is the behaviour that they represent that is the unifying principle, leading to determination of local interaction rules. This presents a shift in the way models are traditionally thought of in the systems biology community, i.e. where conventionally models have been tackled from a mathematical perspective *only* the work presented here stresses the importance of both the mathematical and behavioural perspectives. Taking a behavioural stance does not let the modeller wander outside the realm of what the model should be demonstrating, i.e. providing a connection and context to the real system. It also forces the integrator to think about mathematical or computational components within each model in the context of the whole, and therefore whether some modules even need to be explicated to describe system behaviour (Wolkenhauer, 2001, Chapter 1). The work presented here reinforces ongoing efforts to move towards a true systems-oriented view of biology.

[21] http://www.cancergrid.org

It is of course not claimed here that KDA/ABI is the final solution to the problem. It is claimed, however, that it is an excellent starting point from which modellers can begin to think about integrative measures without impeding freedom of formalism use, as well as address scale and focal issues.

Universality of KDA/ABI

As seen in Chapter 4 and above model integration has been a universal problem and all fields have adapted their own solutions. Note that KDA/ABI, being independent of implementation and using generic components, are impartial to the system that they are simulating. This was a deliberate step in the development of the theories so that their application was not restricted.

The generic nature of KDA ensures that any system that is composed of components and behaviours can be described by it and its properties. Take the classic ant colony example (abstracted from Johnson, 2001):

- Components
    - Ants
        - Types
    - Surrounding objects
    - Predators
    - Food
    - Stimulants/Inhibitions
        - Molecular
- System Concepts
    - Reinforced learning
    - Ant movement
    - Food collection and waste disposal

It is a well-known fact that ants coordinate behaviour of the colony by simple local interaction and hence this is completely amenable to KDA/ABI decomposition.

One could also take the example of traffic, which has received much attention in light of its non-linear behaviour. Here also, components and behaviours can be abstracted from models and implemented in ABI. The very fact that most traffic models are now implemented in ABS anyway reinforces this proposition, in which case further investigation is needed to test whether rule merging can yield plausible integrations.

One limitation however exists where the system cannot be simulated in a closed 3D space. Even if ABI enables linear integrations as explained earlier it fundamentally relies on the fact that components interact locally. There are therefore a subset of complex systems for which KDA/ABI are not immediately suited, most notably financial markets and intelligent social networks, though of course the software system is adaptable. The current system is assumed to be representative of interactions in a spatial context, however interactions between agents and lattice sites could easily be interpreted in a more abstract context.

Other Limitations

Some limitations have already been discussed with respect to the GA and KDA/ABI methods. Judging from the literature review there is a contingent of models for which KDA/ABI will not be able to reconcile in an integration exercise.

Firstly this is the case for stochastic models. Stochasticity is a paradigm as outlined in §4.3.1.1 however it need not always impede KDA/ABI as illustrated by the solution of the Anderson & Chaplain model. Decomposition of rules becomes difficult for stochastic models for 2 main reasons:

1. A stochastic approach yields non-replicable results, which impedes the fitness function's ability to consistently find good/bad performers, which of course has a direct impact on selection pressure.

a. An alternative method could be to include a fuzzy classifier that assesses fitness qualitatively.

2. Equivalently, if component rules are made stochastic the results could potentially be non-replicable.

As suggested above the best solution may be to find a different optimisation method (§8.3.2.1) or fitness function method that can account for diversity in results and assess fitness correctly.

Secondly the combination of centrality and stochasticity could similarly cause problems. A good example is that of Bayesian networks: the simulation must perform such that all state combinations of the network (if that is even computable) are concordant with simulation states.

8.3 Software Appraisal

In this project a parallel computing software infrastructure was created and implemented in Java to enable the model integration strategy described in §5.3. For maximum benefit a fast and reliable platform was needed to conduct both simulations and optimisation algorithms. The software created here proved sufficient in achieving this aim, though there is room for improvement.

The framework design chosen contrasts with other projects where significant changes would need to be made to achieve the desired simulation environment. For example, MASON defines a number of classes for discrete/continuous 1-3D simulations. However to generate hybrid agent-based lattice structures with generic implementations of neighbourhoods, tessellations, etc., which would be necessary in the ABI strategy, would require a significant amount of reprogramming and restructuring. Moreover many of the open-source Java simulation packages would require additional effort to enable simulations over the test cluster.

Of course the drawback to such a well-structured object-oriented framework is one of memory efficiency. Java enables limited control over memory usage – "garbage collection" is by and large automated. However, such issues only become hindrances when memory footprints are so large that the JVM overflows, causing `OutOfMemory` errors – provided that heap sizes are set to sufficient quantities the memory issue should only slow simulations down rather than cause crashes. At any rate, as the clusters running such simulations become larger and the hardware itself advances these problems should become less problematic.

8.3.1 Scalability

The scalability test, i.e. simulating the diffusion model, was designed as a benchmark to determine not only the limits of simulations that would be practical to execute in this project but also whether any simulations made on the Beowulf cluster would be scalable to the larger clusters available at the university. In the 3D lattice initialisation used for the test the total time would be determined by the number of lattice sites. More atomically, this includes the number of interfaces across different machines (limitations incurred by communication over the network) and number of states and rules in each lattice site; additionally with each lattice site there is a cumulative housekeeping load. As the number of lattice sites increases the time taken is therefore expected to be linear. As Figure 6.16 illustrates the scalability of the simulation is not quite linear though the shallow curve is actually quite encouraging. The control simulation showed that the housekeeping load was relatively minor, barely increasing with lattice number. Extra load was not incurred by increased communication via lattice interfaces either, which was shown to be linear (Figure 6.17). It was interesting that scalability was only more than linear once the free memory in the JVM fell during simulations, hence invoking the use of swap. Indeed the preliminary simulations executed on the cluster at the beginning of project, when the nodes each only had 128MB of RAM, behaved similarly though this was not investigated thoroughly. Though this experimentation was for a proof-of-concept this result indicates the limitation of such simulations in that low performance platforms can increase simulation times because of bottlenecks that can otherwise be avoided. This highlights another performance-related limitation of the strategy for deployment, as illustrated in

Figure 6.4, in that the entire simulation can only be as fast as the slowest performing node. Without adopting an entirely new strategy a solution to this is not forthcoming.

Even so, it is interesting to project the scale-up of the software as shown in Figure 8.5.

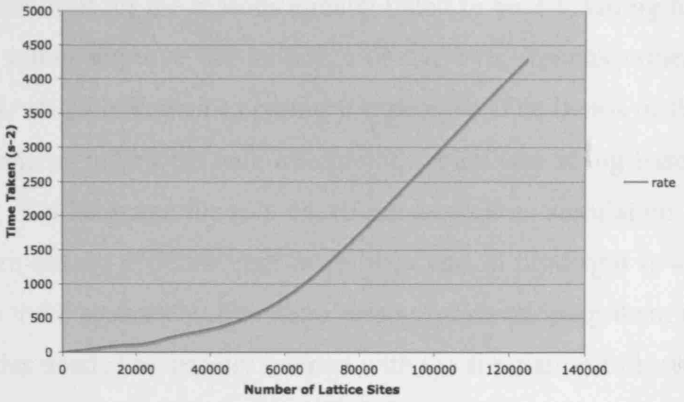


Figure 8.5. Rate of increase in time as lattice number increases. Interestingly the latter two thirds of the curve imply a linear relationship, but in actuality this only counted for three experiments. Clearly more tests need to be performed. However if a linear relationship is assumed, barring caveats such as swapping that affect performance, it would seem for every 60,000 lattice sites there is a cost of 3000 seconds (approx.), i.e. 20 lattice sites per second. Whether inner lattice sites increase load similarly is as yet untested.

8.3.2 Communication Module

The communication test showed a network speed of approximately 25%. The ISO Reference Model[22] explains why, in normal TCP/IP communication, much of the maximum network speed capability cannot be achieved: the standard 7 layers must be traversed at least twice for each packet of data. Adopting UDP[23] instead could theoretically double speed since packet delivery confirmation is not required, though reliability is sacrificed. Some efficiency can also be recovered with the use of newer core Java components: the standard IO components were used to achieve the port-based communication module, however Sun now recommends `java.nio` components that guarantee scalability and higher performance (though there are no published statistics that are relevant to the kind of configuration used in this project). An even better

[22] http://standards.iso.org/
[23] http://tools.ietf.org/html/rfc768

improvement would be integration with industrial-standard software specialising in high volume communication packages, currently the most popular being Tibco RV[24].

8.3.3 Message Passing Interface

A custom MPI was built for the reasons already listed in §6.2.1. Going forward, it would seem prudent to either improve the language or use byte-streams rather than string-to-byte serialisation, which is known to be more expensive. The bonus of the language was that it could be interpreted by the rule interpreter, which was string-based. This allowed the development of a language for rule execution as well as simulation communication, however these are indeed different responsibilities and in hindsight it would have been better to develop these separately. The main reason for developing them together was the fact that some rules needed to directly interact with the simulation infrastructure for some operations, e.g. cloning an agent and serialising it over the network to transport it to another simulation. This could have been achieved with special rules which would have been more efficient since there would be one less communication between the nodes.

8.3.4 Multi-scale Modelling

A metamodelling approach was used to produce scalable models that took into account molecular movement, cellular activity and global behaviour all at once. The metamodel solution is of course one of many recourses that could have been taken in development of the software (discussed in next section) however, admittedly, more must be done to ensure this work can be used practically in tumour modelling. The first step would be finer granularity on the lattice scale, i.e. more lattice sites within lattice sites, which would (i) involve further software optimisation and (ii) more hardware resources.

There are tantalising possibilities in this respect that need further exploration. In particular, various forms of neural networks (NNs) have been presented in the simulation literature that exhibited the ability to interpolate model behaviour (Arndt *et al.*, 2005). Chin & Biondo (2002) successfully approximate non-linear behaviour with multiple function NNs by metamodelling finer-grained systems of DEs and demonstrate their

[24] http://www.tibco.com/

technique on underwater acoustics modelling, which is not readily amenable to real time simulation. Back & Chen (1997) show that systems composed of many subsystems can similarly be approximated by recurrent NNs and hence reduce computational load when fine granularity is not needed. More recently Tolman *et al.* (2005) demonstrate model complexity reduction in wave spectra by use of NNs and function modifications. The applicability of these nets in systems biology, i.e. directly approximating complex biological behaviour, is yet to be seen however it is worth noting that on the local interaction level behaviour of agents is comparatively simple. It will be well worth investigating whether mesoscopic behaviour, i.e. small groups of agents, can be approximated and the corresponding NN(s) linearly integrated with ABI components (e.g. by the architecture shown in Figure 8.3). Such an integration will mark a great step towards bridging the scale problem.

Similarly Bayesian networks (BNs) hold great promise in such a situation. Based on known events, or expert knowledge, this stochastic technique can predict the states of a system(s) based on knowledge of other states. Training of both NN and BN models is imperative, however this can be achieved in the same way as rules are optimised in ABI – simultaneous running of the original model. A use case would be, for example, ECM molecular dynamics models integrated with tissue-level models in ABI. In this case it is of course infeasible to model individual molecules. It may also be the case that metamodelling by lattice site states, as has been demonstrated in Chapter 7, does not capture enough granularity. One can then begin to explore the possibility of building metamodels with NN or BN that quickly and efficiently approximate model states (e.g. production/degradation of molecules) and direct their output directly back to defined agent states (e.g. lattice site states). Such a strategy to the best of the author's knowledge has not yet been applied in systems biology and presents an exciting development that could break one of the greatest bottlenecks in systems biology research. This together with the ABI technique could represents an extremely powerful method by which tumour simulations can be made to a degree not yet reached.

Other Enhancements

One of the main components that is still missing in the software described here is a well-developed 3D visualisation environment. The entire development process was intended to be 100% Java and the two APIs that are of immediate relevance are Java3D (no longer supported by Sun) and Jazz3D. With 3D rendering software model integration interpretation would become a far more enlightening task. The only package that is capable of doing this currently is that of Stamatakos' group at the National Technical University of Athens – however as far as the author is aware this is not generally available and is specialised for the models developed by that group.

8.4 Conclusion

In this project the major advancements can be summarised as follows:

- The determination of the nature of models based on a review of cancer modelling literature.
    - This has led to the determination of characteristics of models that highlight the challenges inherent to model integration. These characteristics are model paradigm, focus, scale and scope.
- A review of model integration literature has been given to assess the extent to which these challenges have been addressed, i.e. the state of the art. This has been extended outside of the systems biology field since model integration is a universal problem in all fields that make use of models and simulations.
- A set-theoretic approach has been developed to tackle the challenge of model semantics. This takes a knowledge-oriented approach to model description.
- An integration strategy has been developed that re-implements models within an agent-based system using local interaction as the basis on which behaviour occurs.
- A well-developed software infrastructure has been developed to enable the above.

Model integration is a key challenge in systems biology. With such a methodology in hand one can begin to build larger and more representative models of biological systems. The solution presented in this thesis tackles all of the identified challenges and, with the suggested improvements, provides a powerful approach to developing an *in silico* tumour that can yield behaviour that has real clinical relevance. Moreover the fruitful start that has been made with software development will give these ideas a head start in the field.

Despite millions of pounds of research cancer continues to be the UK's biggest killer. The introduction of informatics and modelling has brought about a shift in the way scientists understand the disease and how to combat it. Thanks to the rapid growth in affordable computing power the goal of *in silico* experimentation is becoming ever closer. It is hoped the technology developed in this project will be of value to that cause.

LIST OF FIGURES AND TABLES

Figures

Tables

LIST OF EQUATIONS AND FORMALISATIONS

Equations

Formalisations

LIST OF DEFINITIONS

LIST OF ABBREVIATIONS

AI	Artificial intelligence
ABI	Agent-based integration
ABM	Agent-based model
ABS	Agent-based system
AI-IME	Artificially intelligent integrated modelling environment
AMF	Application management framework
API	Application programming interface
BIT	Behavioural inclusion tree
BN	Bayesian network
CA	Cellular automata
CAS	Complex adaptive system
CMDL	Composite Model Description Language
CML	Chronic myeloid leukaemia
COINS	Comparison and integration shell
CPU	Central processing unit
CS	Complex system
CTL	Cytotoxic T-lymphocyte
CViT	Centre for development of a virtual tumour
DAE	Differential algebraic equation
DARPA	Defense Advanced Research Projects Agency
DBMS	Database management system
DCL	Diagrammatic cell language
DE	Differential equation
DEVS	Discrete events simulation
DSS	Decision support system
EC	Endothelial cell
ECM	Extra cellular matrix
ECM	Extracellular matrix
EGF	Endothelial growth factor
EGF	Endothelial growth factor
EGFR	Endothelial growth factor receptor
EPC	Endothelial progenitor cell
EPC	Endothelial progenitor cells
ERATO	Exploratory Research for Advanced Technology
EVD	Effective vessel density
FLAME	Flexible agent modelling environment
GA	Genetic algorithm
GC	Garbage collection
GF	Graphics framework
GUI	Graphic user interface
IECA	International *E.Coli* Association
IMA	Integrated modelling architecture
IME	Integrated modelling environment
ITP	Individual transfer protocol
IV	Immature vessel
JAR	Java archive (file)
JDK	Java development kit
JIT	Just-in-time (compiler)
JRE	Java runtime environment
JVM	Java virtual machine
JWS	Java web simulation
KDA	Knowledge-driven approach
LANL	Los Alamos National Laboratory

LB	Lattice Boltzmann
LG	Lattice gas
LINPACK	Linear algebra software package
LQ	Linear quadratic
MASON	Multi-agent simulation toolkit
MM	Michaelis-Menten
MMP	Matrix metalloproteases
MMS	Model management system
MPI	Message-passing interface
MRM	Model run manager
MS/OR	Management systems and operations research
MSI	Multi-simulation interface
MTS/MCTS	Multicellular tumour spheroid
MV	Mature vessel
MVC	model-view-control
NCI	National Cancer Institute
NCRI	National Cancer Research Institute
NK	Natural killer (cell)
NN	Neural net
ODBC	Open database connectivity
ODE	Ordinary differential equation
OER	Oxygen enhancement ratio
OOP	Object-oriented programming
PBN	Probabilistic Boolean networks
PD	Patch dynamics
PDE	Partial differential equation
PM	Proliferation/migration (decision)
RAM	Random access memory
RBS	Rule-based system
RC	Rule combination
RDF	Resource description framework
RMI	Remote method invocation
SBML	Systems biology mark-up language
SBW	Systems biology workbench
ScriMPI	Scripted message passing interface
SDE	Stochastic differential equation
SM	Structured modelling
SMPI	Short message passing interface
SN	Semantic network
SOA	Service oriented architecture
SOA	Software-oriented architecture
SOAP	Simple object access protocol
SPMD	Same Program Multiple Data
SVM	Support vector machine
TAF	Tumour angiogenic factor
TCP	Tumour control probability
TCP/IP	Transmission control protocol / internet protocol
UCL	University College London
UDP	User datagram protocol
UML	Unified modelling language
XML	Extensible modelling language

APPENDIX A:
Literature Review Results

A1: Model Paradigms, Focus, Scope and Assumptions

Survey of models in the tumour modelling literature has been restricted to the following few on the basis of diversity in paradigm, focus, scope and assumptions. Paradigm profiles, FQCSED, are as follows:

Symbol	Symbol (in main text)	Meaning
F0	$\neg f$	Non-functional
F1	F	Functional
Q0	$\neg q$	Qualitative
Q1	Q	Quantitative
C0	$\neg c$	Discrete
C1	C	Continuous
S0	$\neg s$	Deterministic
S1	S	Stochastic
E0	$\neg e$	Decentralised
E1	E	Centralised
Dx	D	x-Dimensional

Anderson & Chaplain (1998)
Paradigm: F1Q1C1S0E0D2/1, F1Q1C0S1E0D2/1 (alternative solver)
Focus/Scope: Endothelial cell dynamics, vascular dynamics, TAF/Fibronectin dynamics
Assumptions/Context: 3 entities identified as initial players: EC, TAF, fibronectin; continuum model (later discretised) – EC proliferation not considered; motion of EC influenced by: random motility, chemotaxis (TAF concentration), haptotaxis (fibronectin concentration); assume chemotactic sensitivity decreases with increased TAF concentration – some models chose this to be constant; EC birth rate (death rate not considered because of long EC life); simple uptake function; cells do not move out of the universe of discourse; parameters from experimental data; initial condition – circular tumour (0.1mm radius), TAF profile;
Input: TAF and fibronectin terms (production, diffusion and decay rates); EC density per unit area; random motility of EC;
Output: 2D simulation; EC density per unit area, TAF concentration etc.

Arakelyan (2002)
Paradigm: F1Q1C0S0E0D0
Focus/Scope: ODE and Boolean network non-spatial model that takes into account many factors but unique because it also takes into account vessel maturation and destabilisation of mature vessels. Includes the model as an evaluation of anti-angiogenic therapy. Six major processes included: tumour cell proliferation and death, immature vessel formation and regression, immature vessel maturation and destabilisation. Too many components to list but no inputs/outputs as such – Boolean network model (arcs represent Booleans and nodes represent value of variables – ODE models describe the actual values but the network determines the course of the simulation).
Assumptions/Context: 6 processes – tumour cell proliferation, IV formation/regression, IV maturation/mature vessel stabilisation all operate on 3 levels – molecular, cellular, macroscopic (vessel density, tumour volume); proliferation is dependant on vascular density (nutrients) and genetic cell types; tumour growth and vessel maturation go via the relevant regulatory proteins;
Input: Effective vascular density
Output: Immature vessel volume, effective vessel densities, mature vessel volume, tumour size (+ all the other components that describe each node in the Boolean network are outputs)

Arciero et al. (2004)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: growth/response dynamics; immune interaction/siRNA treatment dynamics.
Assumptions/Context: (see inputs below)

Input: antigenicity of tumour; inhibitory factor of TGF-β; death/birth rate of effector cells; max rate of birth/death of effector cell; half-saturation constants; growth rate of tumour (logistic); production/decay rates of IL-2 and TGF-β;
Output: number effector cells, tumour cells; IL-2 and TGF-β concentration;

Bertuzzi & Gandolfi (2000)
Paradigm: F1Q1C1S0E0D3
Focus/Scope: tumour growth dynamics; age dynamics; nutrient dynamics;
Assumptions/Context: tumour chord (given radius), cylindrical symmetry; type of cells – proliferating (inc. cell age) and quiescent (necrosis not considered); simplified cell cycle (all cells go through cycle at the same time) – given times for cell phases etc.; environmental conditions affect transition into quiescence; cell flux independent of cell age or proliferating/quiescent status (described by single velocity field that goes out radially); diffusion of cells not considered; constant cell volume density (volume occupied by cells per unit volume); total cell density throughout chord is constant;
Input: Vessel radius, cord radius; cell flux (velocity) function; proliferation/necrosis/quiescence functions; cell age; oxygen concentration (in blood and at boundary of necrotic tissue), oxygen diffusion coefficient, permeability coefficient of the vessel wall; Michaelis Menten-like oxygen consumption function (including associated constants);
Output: Age density; number of cells; cell flux; oxygen concentrations;

Borkenstein et al. (2004)
Paradigm: F1Q1C0S1E0D3
Focus/Scope: tumour growth and radiation response and angiogenesis.
Assumptions/Context: (see inputs below)
Input: cell status in terms of age, oxygenation and intrinsic radiosensitivity (simulation starts with one cell), apoptosis, cell cycle phase and duration (and subsequent delay due to repair; probabilities for lethal and sub-lethal damage and cell displacement (random walk) are modelled by Monte Carlo method; capillary density, diffusion radius of TAFs, capillary response times (to TAFs), resorption times (duration) of dead cells; oxygen enhancement ratio, radiosensitivity;
Output: corresponding 3D tumour dynamics.

Catto et al. (2003)
Paradigm: F1Q1C0/1S0/1E1D0
Focus/Scope: tumour host characteristics, molecular profile (no dynamics)
Assumptions/Context: Inputs mainly contribute to prognosis to the exclusion of other parameters.
Input: Tumour grade, stage, gender, age, smoker/non-smoker, previous cancer history, p53 status, hMSH2, hMLH1.
Output: time to relapse.

Charasunti et al. (2004)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: Crk-1 pathway & response to Gleevec dynamics. Compartmental model of flow of Gleevec from injected site to blood to interperitoneal cavity and cell.
Assumptions/Context: (see inputs below)
Input: kinetic constants for all biochemical rate equations; synthesis rate of Crk-1 and Bcr-Abl from transcription and translation; background phosphorylation rate of Crk-1; mass of gleevec in interperitoneal cavity, blood and cell; injection rate of Gleevec;
Output: concentrations (nanoMolar) of Bcr-Abl, Bcr-Abl-ATP, 2(Bcr-Abl-ATP), phosphorylated Bcr-Abl, Bcr-Abl-ATP-Crk1, Bcr-Abl-G, phosphorylated Bcr-Abl-G, Crk1, Crk-1 phosphorylated; background phosphorylation rate of Crk-1;

Chignola et al. (2000)
Paradigm: F1Q1C1S1E0D0
Focus/Scope: Gompertzian model of multicellular spheroid growth.
Assumptions/Context: Gompertz curve fitted to data (get constants out) – associated fitting algorithm (solver); introduction of stochastic variable to account for white noise;
Inputs: α and β constants, initial volume.
Outputs: tumour volume.

Cho & Wolkenhauer (2003) and Wolkenhauer (2004)
Paradigm: F1Q1C1S1/0E0D0
Focus/Scope: generalised mass action (GMA), rate equations, the chemical master equation (CME) and Gillespie algorithms to solve molecular dynamics.
Assumptions/Context: Only molecular species with associated parameters considered.
GMA (ODE model):

Input: reaction channels, stoichiometric coefficients, rate constants
Output: amounts of species involved.
Chemical Master Equation (stochastic):
(Can't be simulated directly – have to perform a Taylor expansion to form a Fokker-Planck equation)
Input: probability functions parameters (stochastic kinetic constants for each reaction); 'propensity' of a reaction channel (propensity = the probability of a state change in terms of molecule number in the next time-step); number of species and reactions;
Output: attached probability functions that give probabilities of species S being at amount X at time T.
Gillespie Algorithm:
(Accurately and efficiently simulates the CME – discretised)
Input: propensity for all elementary reactions at each time step; stochastic rate constant that depends on physical molecular properties; stochastic rate constants for reactions; number of distinct molecular combination possibilities.
Output: probability of whether a reaction will occur due to collision of molecules → attached probability functions that give probabilities of species S being at amount X at time T.

Dasu et al. (2003)
Paradigm: F1Q1C1S0E0D2
Focus/Scope: oxygenation (diffusion-limited and perfusion-limited hypoxia) and growth.
Assumptions/Context: oxygen consumption rate assumed to be consistent with Michaelis-Menten-like model; intravascular distances predetermined by Monte-Carlo method (vessels that cross the plane of interest are not counted), i.e. vascular structure is first "grown"; differing oxygen tensions in vessels according to surrounding tissue (in reality differing tensions exist for different vessels for same tissue);
Input: Diffusion coefficient; concentration of gas; consumption-of-oxygen function of cells (includes oxygen partial pressure, pressure when consumption is at half of maximum, maximum consumption rate – all derived from a Michaelis Menten-like equation);
Output: 2D simulation; distribution of oxygen pressures as a result of diffusion and consumption; cell death not considered; simulation starts with 1 tumour cell amongst many normal cells;

De Pillis & Radunskaya (2000)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: competition model for tumour growth (logistic model) with chemotherapy agents and immune cells.
Assumptions/Context: immune cell growth – goes up by presence of tumour and destroys tumour cells by kinetic process; competition between normal and tumour cells for resources; homogeneous tumour; constant influx of immune cells;
Input: influx of immune cells (constant); death/birth rate of cells; amount of drug and relative cell kill;
Output: Number of immune, cancer and normal cells;

Dormann & Deutsch (2002)
Paradigm: F1Q1C0S1E0D2
Focus/Scope: molecular (nutrient) dynamics, cell and cell cycle dynamics.
Assumptions/Context: 2D avascular; other models make assumption of layers within the spheroid, but this model makes no such discrete assumption; assumed that migration of cells is due local pressure and a chemical signal emitted by cells when they become necrotic; each lattice site is only allowed to have one cell; mitosis (only occurs if there's an unoccupied neighbouring site), necrosis, apoptosis all depend on nutrient concentration (thresholds); adhesion, pressure (*mimicked* by movement of cells to areas of low density), chemotactic motility (based on observation of cell movement *in vivo*) all contribute to cell movement; density defined as number of cells + 1/3 necrotic cells (because 2/3 are assumed to be bursting); passive motion (path of least resistance) – i.e. no energy required;
Input: nutrient diffusion equation approximation by "moving averages"; rules for cells (apoptosis, necrosis, mitosis) based on local nutrient supply; rules for adhesion, pressure, chemotactic motility;
Output: 2D simulation.

Kansal et al. (2000a)
Paradigm: F1Q1C0S0E0D3
Focus/Scope: cell dynamics; tumour dynamics.
Assumptions/Context: Varying 3D lattice grid with Voronoi tessellation allows simulation to continue over 3 orders of radial magnitude – therefore there is variation (CA) cell density. Nutritional gradient not modelled specifically, but rules for each are given in terms of distance from the edge such that if the distance is beyond certain thresholds then the cell changes its state.
Input: rules for mitosis time (probability of division including modified probability for new strain), nutritional needs of growth-arrested cells, nutritional needs of dividing cells, effects of confinement pressure (maximum tumour extent); maximum number of time steps;
Output: 3D simulation.

Kansal et al. (2000b)
Paradigm: F1Q1C0S0E0D3
Focus/Scope: (as in above model)
Assumptions/Context: Extension of above model. Tumour simulation is as normal except at a certain radial threshold a second "strain" of cell is initiated (randomly out of the proliferating) population that has an altered probability of mitosis.
Input/Output: as above plus maximum number of time steps;

Koch et al. (2004)
Paradigm: F1Q0/1C0S0E0D0
Focus/Scope: sucrose and related metabolic enzyme dynamics
Assumptions/Context: Petri Nets to model sucrose metabolism. Places represent compounds, transitions represent reactions (name of enzyme given to transition).
Input/Output: (see above)

Lakmeche & Arino (2001)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: growth dynamics; dosimetry
Assumptions/Context: cells are sensitive/resistant due to mutation (cell cycle not considered).
Input: Drug dose; growth rate constants of sensitive and drug-resistant cells
Output: biomass of sensitive/drug-resistant cells;

Levine et al. (2001)
Paradigm: F1Q1C1S1E0D0
Focus/Scope: Random walk and Michaelis Menten-like model for angiogenesis (at the level of cells and molecules using enzyme kinetics). Concentrates on the role of fibronectin (i.e. haptotaxis).
Assumptions/Context: (see below)
Inputs: density of receptors on ECs; concentration of angiogenic factors; concentration of proteolytic enzymes (that degrade cell surface receptors and recycles them to active form); concentration of intermediate receptor complex; concentration of ECs; density of fibronectin; probability rate per unit time for movement of EC (including probability density function where an EC can be thought of as a cloud that covers a certain amount of space); control substance term (aggregate effect of angiogenic substances); chemotactic/haptotactic sensitivity;
Outputs: concentration of proteolytic enzymes (that degrade cell surface receptors and recycles them to active form); concentration of intermediate receptor complex; concentration of ECs; density of fibronectin;

Mantzaris et al. (2004) [continuum models]
Paradigm: F1Q1/0C1S1/0E0D0/1/2/3
Focus/Scope: (see below)
Assumptions/Context: (see below)
Input: functions of cell populations (ECs, etc) and concentration gradients of molecules including associated coefficients; flux functions (movement of EC cells); tactic sensitivity (EC response to gradient; functions for proliferation and production/degradation of chemical compounds;
Output: cell population dynamics, molecular concentrations and chemotactic/haptotactic response of ECs.

Mantzaris et al. (2004) [discrete models]
Paradigm: F1Q1/0C0S1/0E0D0/1/2/3
Focus/Scope: (see below)
Assumptions/Context: (see below)
Inputs: stochastic differential equations, damping terms, noise terms for modelling random walk of sprout tips, chemotactic terms; viscosity coefficient; Wiener process; angle of movement of sprout movement relative to other components (e.g. TAF source); proliferation rate (of neo-vessel cells); number of branches of each sprout; redistribution coefficient (of EC from sprout to each of its branches);
Outputs: position of sprout formation; position/velocity of sprout tips; sprout speed/length; vessel density;

McDougall et al. (2002)
[Same as Stephanou (2005) except solved in 2D]

Patel et al. (2001)
Paradigm: F1Q0/1C0S1/0E0D2
Focus/Scope: role of H^+ production (i.e pH) in early tumour growth and angiogenesis.
Assumptions/Context: occupation of area of maximum number of cells.

Input: CA rules for occupation, H^+ concentration, glucose concentration (including diffusion equations); microvascular density (randomly distributed in lattice); glucose consumption rate; acid production rate; flux through vessel wall; permeability and diffusion constants;
Output: H^+ and glucose field; tumour radius; 2D simulation.

Pinho & Nini (2002)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: Six ODE expressions, 3 for a primary tumour site and the same 3 for a secondary site. Terms for interactions between normal cells, cancer cells and chemotherapeutic agents. Logistic growth terms for cells with predator-type terms for action of chemotherapy agent and competition terms between normal and cancer cells.
Assumptions/Context: migration of cancer cells from primary to secondary site is defined as metastasis (concentration of agents given by ODE, inc. concentration of normal, cancer and chemo agents + infusion and washout rates); time delay for when cells start to interact at the secondary site; competition (chemo as predator, cells as prey); inequalities introduced based on assumptions that cancer cells out-compete normal cells regardless of initial conditions & cancer cell growth factor;
Input: concentration of normal cells, cancer cells and chemotherapy agent at primary site and at secondary site (= 6 equations); birth rates, carrying capacities, competition coefficients, predation coefficients (inc. combination rates of therapy agent with cells), infusion rate of therapy, washout rate of therapy; rate of cancer cells leaving primary sight; time delay of metastasis; proportion of "left" cells that reach and interact at second site.
Output: concentration of normal cells, cancer cells and chemotherapy agent at primary site and at secondary site (= 6 equations).

Qi et al. (1993)
Paradigm: F1Q1C0S1/0E0D2
Focus/Scope: immune system and growth dynamics; nutrient dyanamics; invasion (equivalent to tissue flux);
Assumptions/Context: immune-system effector cells (numbers remain constant) action on tumour cells; 2D mesh (simulation starts with 5 cancer cells at centre and other lattice sites occupied by normal and effector cells), one time step = 1 day, 4 nearest neighbours → each lattice site has one effector cell, can have normal/cancer cell (proliferative, dead, cell-effector complex); proliferation probability assumed to be dependant on total tumour size and maximum possible size (max size acts as an encapsulating variable that represents growth restriction, i.e. nutrient and toxicity levels, etc); random invasion of neighbouring sites; parameters taken from literature;
Input: stochastic rules for effector cells consuming cancer cells in a certain number of time steps; stochastic rules for mitosis in certain number of time steps; densities of cells;
Output: 2D simulation.

Reis et al. (2001)
Paradigm: F1Q1C0S0E0D2
Focus/Scope: cell adhesion model (simulated in Voronoi grid) to simulate tumour growth, roughness and detachment. Cell death is not considered.
Assumptions/Context: cell motion defined by cell interactions between cells and mitotic rates in the form of force equations; 2 types of cells – normal, tumour;
Input: Adhesion factor α for each tumour cell (also called repulsion factor) generated randomly for each new cell by a Gauss-Heaviside distribution (all normal cells have same α); time interval between cell divisions; cell radius; standard deviation of the Gauss distribution of the individual cell α's (σ).
Output: 2D simulation.

Scalerandi et al. (2003)
Paradigm: F1Q1C1/0S0E0D2
Focus/Scope: avascular to vascular phase transition in tumour cords. Six species considered: tumour cells, normal cells, necrotic cells, vessel-forming ECs, migrating ECs and fixed ECs. Nutrients and TAF molecules also considered (including diffusion). Volume of cells and resultant stress is also modelled
Assumptions/Context: 2D mesh; species that can occupy sites – cells (cancer, necrotic, healthy), EC (vessel-forming, migrating, fixed – each may transform into each other), molecules (nutrient, TAF); energy assumed to be directed to either metabolism or mitosis (measured as energy – internal, absorbed, consumed); swelling of tissue caused by local stress (due to mitosis and nutrient flow → mass transport); TAF absorption assumed to be similar to nutrient absorption (assumed to be same for all ECs);
Input: rules for cells and molecules; Nutrient/TAF absorption rates; energy consumption terms; rules (according to internal energy of cell) for starvation, mitosis or dormancy; reaction-diffusion equations with associated constants, sink/source rates; cell volume and stress including elastic constant.
Output: 2D simulation.

Schoeberl et al. (2002)

Paradigm: F1Q1C1S0E0D0
Focus/Scope: EGF pathway dynamics;
Assumptions/Context: A + B ↔ AB ↔ C then the forward reactions (reactants to complex and complex to product) can be combined – similarly for the backward reaction. This yield two rate constants (one forward and one backward).
Input: rate constants, Michaelis-Menten constants, turnover numbers, initial conditions.
Output: State (amount of) each entity in the model at specified time.

Schuster et al. (1999)
Paradigm: F1Q0/1C0S1/0E0D0
Focus/Scope: metabolic pathway dynamics (generalised)
Assumptions/Context: that dynamics can be decomposed into elementary modes and is based on stoichiometry.
Input: network structures and stoichiometry of reactions.
Output: molar concentration simulation.

Sherrat & Chaplain (2001)
Paradigm: F1Q1C1S1/0E0D1
Focus/Scope: tumour growth including functions for cell densities of proliferating, quiescent and necrotic cells – all three functions are functions of time and a single x coordinate. Functions for cell movement and generic nutrient also included.
Assumptions/Context: 1D (therefore "assumed to be good for small tumours") ; spheroid morphology; no assumption of sharp division in space between cell types, i.e. no separate compartments of cell types therefore cell densities are modelled; migration assumed to be function of "contact inhibition of migration"; proliferation rate limited by crowding effects of total cell population; transition of cells from proliferating to quiescent to necrotic due to nutrient/growth factor levels;
Input: Associated constants.
Output: Number of proliferating, quiescent, necrotic cells including their positions.

Shmulevich et al. (2002)
Paradigm: F1Q1/0C0S1E0D0
Focus/Scope: Probabilistic Boolean Networks (PBN) to model genetic regulatory networks. Ordinary Boolean network have been used in the past for modelling genetic regulatory networks, where the nodes are ON/OFF switches for specific genes and arcs represent logical rules of relatedness between genes. Nodes therefore have Boolean functions that convert their inputs into outputs (gene is on/off). For the PBN every node now has a set of functions, rather than just one, and each function is assigned a probability at runtime to determine which function should perform a node transition.
Assumptions/Context: (see focus and inputs)
Input: network structure; probability functions;
Output: genes ON/OFF

Stamatakos et al. (2001,2002) & Antipas et al. (2004)
Paradigm: F1Q1C0S0/1E0D3
Focus/Scope: radiotherapy response; growth dynamics; pathway dynamics; cell cycle dynamics;
Assumptions/Context: Spheroid morphology (therefore no angiogenesis); cell cycle (division can occur even if there's no space, and dependent on nutrient and oxygen supply); heterogeneous genetic population; no metastases; side effects; cell response – phase & oxygen dependant response; 3D mesh (100×100×100); nutrient supply surrounds spheroid; only horizontal and vertical communication between cells allowed; simulation starts with 1 cell; cell lysis and apoptosis – products diffuse outwards causing tumour shrinkage (presumably this is because of pressure from the outside-in); least-number-of-cells-shift rule for tumour expansion (should be considered part of a solver); time quantisation – 1 hour time-step; cell cycle duration obeys normal Gaussian distribution;
Input: Oxygen enhancement ratio; detailed model cell cycle (including phases and duration and attached probabilities in each phase for death – inc. probability of cell death because of age and apoptosis); tumour image data; p53 status (wild type or mutated); radiotherapy dose; lack of oxygen → vascularisation (very basic phenomenological model); LQ model for survival probability of a cell (attached α and β values);
Output: 3D Spatial model.

Stephanou et al. (2005)
Paradigm: F1Q1C0/1S1E0D0/2
Focus/Scope: Stochastic terms for directions the endothelial cells (for vessel extension) will go in → transport to spatial mesh. Blood flow modelled by simple Poiseuilles' Law in locality (discretised cell). Therefore this model has a blueprint model that is approximated and then transported to its own solver simulation (spatial mesh). Blood flow through the network is modelled. Blood vessels considered static.

Assumptions/Context: EC random motility, chemotaxis, haptotaxis; non-dimsional; term for decreased TAF sensitivity for increased TAF concentration; Euler method to solve ODE; probabilities for direction of movement of EC; branching – probability of existing sprout to increase branching as TAF concentration goes up (level of maturation must reach threshold before it can branch); anastomosis – if two branches collide, only one is allowed to continue to grow (random choice); if a sprout tip meets another sprout, they fuse to form a loop; EC do not move back to the sprout from which they come; 70×70 grid (square cells); initial fibronectin and TAF concentrations;
Input: Terms for random movement, chemotaxis (of TAFs, inc. uptake by cells term) and haptotaxis (of fibronectin, including production & degradation terms); length, radius, pressure, viscosity (for flow); drug volume in tumour (inc. "filling time" of drug into section of capillary).
Output: endothelial cell density, fibronectin and TAF concentrations; endothelial cell migration (up, down etc. determined by stochastic parameters); flow.

Stewart & Traub (2000)
Paradigm: F1Q1C0/1S0/1E0D0
Focus/Scope: Stochastic intensity-modulated radiation therapy model (plus multibeam radiation therapy) affects tumour dynamics.
Assumptions/Context: (see focus and inputs)
Input: Proposed tumour volume (voxelised), calculation of initial number of cells; Monte Carlo simulation used to calculate delivered dose; Dose rate;
Output: tumour cell kill; tumour control probability

Stott et al. (1999)
Paradigm: F1Q1C0S1E0D2
Focus/Scope: adhesion and growth. Necrotic, quiescent, normal and proliferating tumour cells considered distinct.
Assumptions/Context: nutrients at base of 3D mesh (200×200×200); cells have energies/elasticities with given bounds (membrane distortion, maximum volume); no vascularisation; cell types (including growth rates and volume constraints) – necrotic (binding properties assumed to be the same between all necrotic cells), quiescent, proliferating tumour cell, healthy cell; cell volume to surface ratio affects mitotic rates (thresholded) – therefore growth rate is assumed to be directly proportional to supply of nutrients (increase cell surface → increased nutrient uptake → increased growth); growth of normal cells not considered (birth/death rates assumed equal); normal tissue is a homogenous source of nutrients; nutrient thresholds for proliferation/death;
Input: Growth/decay rates and growth constraints for each cell type; adhesive strength between tumour/normal cells; volume/surface area of cells and centre of mass of cell (in mitosis the cell splits through the centre of mass); nutrient concentrations; membrane elasticity; Nq (critical value below which proliferation cannot occur);
Output: 2D simulation

Stoll et al. (2003)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: angiogenesis model (non-spatial) that takes into the account the affects on circulating progenitor endothelial cells as well as vessel wall-related endothelial cells. Subsequent therapy response.
Assumptions/Context: initial state = small clump of tumour cells and vascular network; compartmentalisation of human body; stimulatory and inhibitory factors assumed to be a single parameter (α); when $\alpha > 0$ → angiogenesis increases (adhesion of EPC to tumour vasculature is enhanced); biodistribution model → circulating EPC;
Input: α (relative imbalance between stimulatory and inhibitory factors); EC density; rate of accumulation of EPCs at site; concentrations of angiogenic growth factors (separate variables for those produced by tumour and those produced elsewhere), (biodistribution model); Blood flow rate and vascular volume.
Output: change in tumour radius over time.

Sung & Simon (2004)
Paradigm: F1Q1C1S0E0D0
Focus/Scope: mass action kinetics of NF-κB and effects of an inhibitor drug.
Assumptions/Context: (see inputs and focus)
Input: network structure; kinetic parameters.
Output: concentrations of molecular species.

Tan & Chen (2000)
Paradigm: F1Q1C1S1E0D0
Focus/Scope: carcinogenesis due to environmental conditions. Takes into account "one-stage" and "two-stage" pathways to carcinogenesis. Three different probability generating functions depending on status of cells converting from normal to cancer or death.
Assumptions/Context: (see input and focus)
Input: probabilities for one/two-stage; initial number of cells;

Output: incidence of tumour forming, given as probability.

Voitikova (1998)
Paradigm: F1Q1C0S0/1E0D2
Focus/Scope: tumour growth dynamics/response to immune interaction. Random walk of cells.
Assumptions/Context: (see focus and input)
Input: initial configuration and occupation rules (cells occupy CA lattice sites); immune elimination capacity; tumour growth rate (+ competition term for resources); immune cell inactivation, influx and elimination; number of effector cells; random walk of immune cells (tumour cells are stationary); state of immune cells – active/inactive, living time, movement orientation; state of immune cell: time till next mitosis (stochastic rules for growth);
Output: 2D simulation

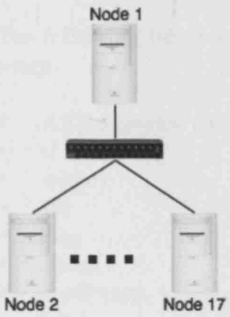
Wascheck et al. (1997), Ammar & Hussein (2003), Hussein & Ahmed (2000)
Paradigm: F1Q0/1C0/1S0/1E1D3
Focus/Scope: none.
Assumptions/Context: imaging data and solver is enough to elucidate locations of tumours.
Input: 3D image data
Output: areas of interest in 3D map

Zevedei-Oancea & Schuster (2003)

Paradigm: F1Q1C0S1E0D0
Focus/Scope: metabolic networks flux
Assumptions/Context: (see input)
Input: node/arc structure including placement of transition and places; Stoichiometry matrix;
Output: dynamics of Petri Net (tokens at places is equivalent to number of molecules, i.e. moles of reactant or product)

APPENDIX B:
Software Results

B1: Beowulf Cluster Specification and Architecture



Star topology connected by DLink 24-port 100MBs^{-1} network switch.
18 IBM PC compatible Pentium III at 866 MHz, 1GB RAM, 20GB Hard Disk, network card, 3½" Floppy drives, CD-ROM drive, 10/100MBs LAN cards.

B2: SMPI and ScriMPI

Background

A language that will pass information between Master and Slave nodes concerning states of Jivas. The API is also used to initialise simulations on remote machines.

The API therefore needs to be able to do the following:

- Encapsulate information about Jivas (can be done through serialization mechanism)
    - Positions (SpaceModelPosition information)
    - Neighbourhood
    - Time index
- Jiva states (can be done through serialization mechanism)
    - Number, Boolean etc.
    - Generic State
- Initialisation information (can be done through serialization mechanism)
    - JivaContainer initialisation information (size, content, etc.)
    - SpaceTimeModel information
- Request neighbours of a specific Jiva and invoke them to perform a full range of tasks.
- Request information about a specific Jiva at a specific time-step.
- Distribution and firing of graph section for linear integration.

Language Details

The language was not meant for human readability, but keeping it short and clear was an obvious goal.

To avoid massive complexity, the language is implemented in Java-like and declarative (like SQL). Through this language one can invoke instantiations and method invocations on a remote machine.

There are two types of messages: (i) a simple message, which elicits no response (ii) a command, which elicits a response. Also, there are two types of language: (i) Short Message Passing API (SMPI) and (ii) the

scripting version (ScriMPI), which uses SMPI with control structures and UNIX binary-like architecture. SMPI is purely used as a communication device between machines in a cluster whereas ScriMPI is used to actually coordinate how a simulation proceeds. SMPI is always a one-line command with a new line character on the end (serial execution).

Each command/message can either be sent from slave to master or master to slave, which will affect the way the command/message and respective switches are used.

Reserved Characters and Words

The following words must not appear in any command or script apart from designated SMPI or ScriMPI usage:

- All designated commands/messages
- if
- while
- for
- else
- failed
- confirmed
- $index

The following characters must not appear in any command or script apart from designated SMPI or ScriMPI usage:

- colon and semicolon
    - ;
    - :
- ampersand
    - &
- dash
    - –
- brackets
    - ( )
    - { }
    - [ ]

Basic Syntax

command –switch parameters

"Switch" part can be any length String, so long as the interpreter recognises the switches coupled with the command. A space must exist between the command the "-" and no space is needed after that for the switches. A space must exist for every switch. Switches are optional.

Multiple Switches and Parameters

More than one switch is allowed for a single command/message. However, each switch might require a parameter. The message parser for SMPI will use strict ordering to resolve parameters and switches. If a command (e.g. *command*) is combined with one or switches (e.g. *a* and *b*) then the parameters for each switch must be given in the order that the switches appear and each parameter must be separated by a semicolon:

command –ab param_a ; param_b

Variable Declarations

Always done with an equals sign. All variables begin with a $ sign.

$var = ...

Whenever one refers to a variable in later calls the $ sign must be present.

Declarable variables are: String, integer (short, normal and long), floating point, Boolean.

Declaring a String:

$var = "foo" (always put in quote marks)

Declaring an integer (4 and 8 byte):

$var = 67i (4 byte)
$var = 67l (small L) (8 byte)

Declaring floating points (4 and 8 byte):

$var = 67.543f (4 byte)
$var = 67.543d (8 byte)

To declare the object versions of primitives (Float, Double, etc.) use capital letter, e.g. 67I, 67L etc.

Declaring Booleans:

$var = true
$var = false

Reassigning variables:

$var1 = ...
$var2 = $var1 *($var2 is now a reference to $var1)*

Additionally the part following the equals sign need not be just a number, string or Boolean. It can also be a complex programmatic line that includes other commands, but one must be careful that such commands return some value since that value is to be assigned to the variable.

Basic Commands/Messages

Designated message: `confirmed`
Description: confirms a specified event to the server.
Parameters: Leading String after the command denotes what the confirmation is.
Switches: none.

Designated command: `npp server:client`
Description: sets a new server/client port pair (new port pair).
Parameters: server, integer server port; client, integer client port
Switches: none.

Designated command: `killsock`
Description: kills the socket connection at the remote machine.
Parameters: none.
Switches: none.

Designated command: `mcall methodName`
Description: calls a specific method. As an SMPI message, the user is restricted to the name of a single method and cannot specify any parameters for the method. The method must exist in the MPIConnector class that handles message passing (e.g. a child of modelomics.core.server.MMaster).
Parameters: Leading string is the exact name of the method to be called (case sensitive).
Switches: none.

Designated command: `exitvm -[a]`
Description: terminates the Java virtual machine.
Parameters: none.
Switches: *a*, filters message to all workhorses (simulation completely shuts down).

Designated command: `faw smpimessage`
Description: (for all workhorses) Executes the given parameters on all registered workhorses. NOTE: that is only meant to work on the local machine if the local machine is running Modelomics and NOT ModelomicsWH; if it runs on ModelomicsWH, the *immediate* parental master node is sought and the SMPI message is sent all nodes (NB: when a workhorse calls *faw* and a Master is found, it is not necessarily the topmost Master!).
Parameters: *smpimessage* – an SMPI message as would be written without the faw.
Switches: none.

Designated command: `sendM smpiMessage`
Description: Sends a message to the master server (can only originate from a workhorse)
Parameters: smpiMessage, the message that needs to be sent.
Switches: none.

Designated command: `sendC ipHost smpiMessage`
Description: Sends a message to the specified client machine.
Parameters: ipHost, the IP/host address of the client to which the message should be sent; smpiMessage, the message that needs to be sent
Switches: none.

Designated command: `sendTF fileName`
Description: requests the receiving machine to send the contents of the specified text file over the local buffer system with the ID as the whole message. The sent object is a serialised java.util.List<String> object.
Parameters: *fileName*, file to send.
Switches: none

Designated command: `ls fileDir`
Description: requests the receiving machine to send the contents (i.e. full file paths only) of the file directory over to the local buffer system with the ID as the whole message. The sent object is a serialised java.util.List<String> object.
Parameters: *fileDir*, directory whose contents are to be sent.
Switches: none

Designated command: `runp -[sj] commands`
Description: Runs the given commands on the receiving machine in a separate process.
Parameters: *commands*, commands to run in a separate process.
Switches: s, runs this process in a separate process from the current one (i.e. the current thread does not wait for the new process to finish); j, runs this process in the same process as the current one (i.e. the current thread wait for the new process to finish)

Designated command: `write -[an] fileName line`

Description: writes the given line to the given file (created if it doesn't exist already).
Parameters: *fileName*, file to write to; *line*, line to write
Switches: a, append the file with this line; n, create a new file with this line (do not append).

Designated command: `wsmem semanticCode value`
Description: goes through the JivaContainer in the local buffer and sets any Memorable- and Specie-implementing objects with the given semantic code to the given value
Parameters: *semanticCode*, the semanticCode to which the value applies. A more generic code can be supplied by putting it in square brackets, which will force the code to be parsed as a regular expression; *value*, the value to which the object is set (must be "true" or "false").
Switches: none.

Designated command: `rm -[chijlnop] id`
Description: instructs the local buffer to remove a particular item from itself if it exists.
Parameters: Leading String is an ID for the object that the client needs to remove. If a switch is not provided, then the string following the rm command must be a key for the buffer (object class type).
Switches: ONLY ONE SWITCH IS ALLOWED, and is optional. c, request for a sts.JivaContainer; h, request for optim.ga.Chromosome; i, request for optim.ga.Individual; j, request for sts.Jiva; l, request for list of optim.ga.Evaluatable; n, request for sts.Neighbourhood; o, request for generic object; p, request for optim.ga.Population;

Designated command: `extract ip key id`
Description: instructs the receiving machine to extract an object from the buffer of a remote machine (that object is permanently removed from the remote buffer and placed in the local buffer with the given key and id). The *ip* is only necessary when this message is being sent to a master, which needs to know which slave machine to get the object from. Otherwise it is assumed the object is to be retrieved from the master. See localext.
Parameters: *ip*, the IP or hostname of the remote machine; *key* and *id*, the unique identifiers for the object in the remote buffer.
Switches: none.

Designated command: `localget key id`
Description: Gets an object from the local buffer. Object can be captured:

$obj = localget key id

Parameters: *key* and *id*, the unique identifiers for the object in the local buffer.
Switches: none.

Designated command: `localext key id`
Description: Extracts an object from the local buffer (i.e. the original object is removed from the buffer permanently). Object can be captured:

$obj = localext key id

Parameters: *key* and *id*, the unique identifiers for the object in the local buffer.
Switches: none.

Designated command: `runj id`
Description: Runs a job that is in the buffer
Parameters: *id*, the unique identifier for the object in the local buffer.
Switches: none.

Designated command: `testcomm`
Description: tests communications and outputs to standard out and console

Parameters: none.
Switches: none.

Initialisation Commands/Messages

When Modelomics starts running on the Master machine, it needs to a pass messages to specified nodes and instruct to get a reference stub of the server object.

Designated command: `sconnect hostNameOrIP`
Description: tells recipient to connect to server
Parameters: Host name or IP address to connect to.
Switches: none.

Designated command: `simready`
Description: tells server simulation control object that this client is ready to start a simulation. This can only be sent from slave to a master.
Parameters: none.
Switches: none.

Designated command: `uprule -[sf] rule sn rulesetID rulesetName`
Description: updates rules for jivas.
Parameters: rule, the rule as a string or a file location to read a rule(s) from; sn, the semantic code to which the rule apply; rulesetID, ruleset's ID; rulesetName, ruleset's name.
Switches: s, parses parameter rule as a rule string; f, parses parameter rule as a file location.

Designated command: `status -[v]`
Description: requests information from all clients pertaining to their simulation status
Parameters: none.
Switches: v, verbose output (prints status to console if it exists)

Designated command: `detn`
Description: requests receiving client's primary container to immediately determine it's neighbourhood and to fix that neighbourhood from that point on. This should only be used when neighbourhoods are intended to be fixed (e.g. in Lattice models).
Parameters: none.
Switches: none.

Designated command: `get -[chijlnop] id clientIP`
Description: instructs the recipient machine to collect a specific item.
Parameters: Leading String is an ID for the object that the client needs to retrieve. If the message has been sent from a client to a server then the client additionally needs to provide its IP address. If the client has been supplied as an IP to the master machine that is running Modelomics, then the clientIP variable must be given as exactly that IP (giving the equivalent hostname will NOT work, and vice versa).
Switches: ONLY ONE SWITCH IS ALLOWED. c, request for a sts.JivaContainer; h, request for optim.ga.Chromosome; i, request for optim.ga.Individual; j, request for sts.Jiva; l, request for list of optim.ga.Evaluatable; n, request for sts.Neighbourhood; o, request for generic object; p, request for optim.ga.Population;

Designated command: `start`
Description: starts or continues the simulation
Parameters: if switch is given, leading String represents discrete/continuous time for simulation to run.
Switches: none.

Designated command: `halt`
Description: halts the simulation but does not kill it.

Parameters: none.
Switches: none.

Designated command: `stop`
Description: kills the simulation.
Parameters: none.
Switches: none.

ScriMPI-specific Commands, Messages and Syntax

ScriMPI is a script (can be in the form of a plain text file) that is parsed line by line and encapsulated in a modelomics.util.Job object. One Job can be used only in relation to a single object. ScriMPI can contain any of the SMPI commands/messages and can also include the following.

Designated command: `mcall $var_obj methodName ($var_1…)`
Description: Advanced version of the SMPI mcall command. The command is not just restricted to a single class but can be applied to any class and can take any number of parameters. The return value of the method can be captured by a simple declaration:

```
$return_value = mcall $var_obj methodName ($var_1,…)
```

Parameters: $var_obj, the object (variable) that has the specified method; methodName, the name of the method to be called; ($var_1…n) the explicit parameters the method needs to be executed. If there are no explicit parameters for the method the brackets should be left empty. Note that a space must exist between the method name and the first bracket. Note that no spaces are allowed between the individual parameters inside the parentheses. Note that each parameter in the parentheses are separated with commas. Variables arguments in methods are only supported if no other parameters exist.
Switches: none.

Designated command: `execute -v fileName`
Description: Executes the script given in fileName. Note that all file separators will be automatically converted to the current platform separator but by convention should use the UNIX separator, "/". All variables that start with $trans_ are regarded as temporary (transient) variables. Once a script has finished running using the execute command, any variables labelled as transient are not captured.
Parameters: name of file that needs to be executed.
Switches: v, capture variables. Warning: by using the v switch, the named variables in the named file come into the users' local space. Therefore if any of those variables are not unique to the space the original variables could be overwritten. To avoid this, make sure all variables are unique both in the local space and in the named file.

Designated command: `buffer $variable uniqueID`
Description: Puts the variable in the local modelomics.util.Buffer.
Parameters: uniqueID, the ID of the variable.
Switches: none.

Designated command: `cconnect hostNameOrIPOrIPRange`
Description: tells the master server to establish a new client. An entire range of clients can be initialised if the following syntax is used:

```
cconnect a.b.c.x:y
```

Note the colon that denotes the range x to y. Note that there are no spaces other than the one between the command and the parameter. If this command originates from a client, the request is sent to the master server.

Parameters: Host name or IP address/IP range to connect to.
Switches: none.

Designated command: `rnj $vicinity jivaContainerID localBufferID`
Description: (request neighbourhood for Jiva) requests the server (from local client) for neighbourhood information pertaining to the given vicinity. The result leaves out any Jivas in the given JivaContainer and is put in the originating clients' buffer with the given ID.
Parameters: vicinity, the vicinity in question (variable MUST be an sts.Vicinity object); jivaContainerID, the JivaContainer the originating Jiva is currently in; localBufferOD, the desired ID that the results will be put in (retrieve by getting Neighbour object from buffer with this ID).
Switches: none.

Designated command: `sharer`
e.g. `$var = sharer`
Description: Assigns the local ModelomicsSharer to the given variable. Therefore this enables the user to gain access to handle on the local sharer object (Server and/or Client). Note "Local" here means the sharer that is *receiving* the message.
Parameters: none.
Switches: none.

Designated command: `store $var ID`
e.g. `store $pop myPOP`
Description: Stores a variable in the script executor's internal map for retrieval.
Parameters: $var, variable to be stored; ID, unique string identifier.
Switches: none.

Designated command: `mpc`
e.g. `$mpc = mpc`
Description: Sets the given variable to the MPIConnector associated with the machine that is running the current script.
Parameters: none.
Switches: none.

Designated command: `hostip`
e.g. `$h = host`
Description: Finds out the current IP for this machine and sets the variable to that IP or hostname
Parameters: none.
Switches: none.

Designated command: `pj $jiva_container`
Description: puts the given variable (which must be of type sts.JivaContainer) into the local buffer as the *primary JivaContainer* for a simulation
Parameters: $jiva_container, the JivaContainer that needs to be stored.
Switches: none.

Designated command: `rrm`
Description: Requests the local RuleBasedSystemControl for a RuleModel object. This can be captured by the following syntax:

```
$rm = rrm
```

Parameters: none.
Switches: none.

Designated command: `rbsc`

Description: Requests a reference to the local RuleBasedSystemControl. This can be captured by the following syntax:

```
$rm = rbsc
```

Parameters: none.
Switches: none.

Designated command: `echo $var`
Description: prints out variable values to the standard output stream
Parameters: $var, variable that should be printed out
Switches: none.

Designated command: `concat $var...`
Description: Concatenates all the declared variables into one string. The string can be captured:

```
$con = concat $var1,$var2,...
```

Note: there are no spaces after the first variable and all variables are separated by commas.

Parameters: $var..., variables separated by commas
Switches: none.

Genetic Algorithm-related Commands

Designated command: `gaexe`
e.g. `$ga = gaexe`
Description: Sets the given variable to the GAExecutionModel associated with the machine that is running this script.
Parameters: none.
Switches: none.

B3: GA Attributes

Configuration File Format

- Tab delimited (key/value)
- Lines starting with # are ignored

Jobs can be put into this file the same way as they can be put into master/slave configuration file. Jobs labelled with an ID that starts with "init" are executed in the same way before the execute method in GAExecutionModel fires. This is so that if there are any specialised implementations of any of the GA models with (for example) methods that are not defined in the interfaces that need to be called before the GA runs, one may perform this action by putting it into an init job.

The GARun class is a main class that initialises the GA using the ga.prop file. The GARun class undergoes the following procedures in precise order:

1. Reads ga.prop file, extracting all relevant information.
2. Initialises all models (defaults are the String-based models in the sts.ga package).
3. Sets all models to a GAExecutionModel-implementing class (also defined in property file).
4. Initialises a GAPopulation (defined in ga.prop as Jobs).
5. Invokes the execution of a GA via the GAExecutionModel in 3, passing the GAPopulation in 4 as a parameter.

Configuration Attributes

`cprob-within x`
Mechanism in DefaultGAExecutionModel is to check the xxx value for cprob-within to determine whether a chromosome should cross within or across different individuals. If cprob-within is set to 0, then the execution model will ensure that all crossovers will occur between different individuals.

`icsel x`
Crossover selection rate. Probability that an individual is picked for a crossover operation.

`chcons x`
(cross whole consequents)
Defines probability of whole consequents being crossed if chromosomes have already been selected for crossover.

`chconsi x`
(cross whole inner consequents)
Defines probability of whole inner consequents (separated by ;;) being crossed if chromosomes have already been selected for crossover.

`chcond x`
(cross whole conditions)
Defines probability of whole conditions being crossed if chromosomes have already been selected for crossover.

`chb x`
(cross elements)
Defines probability of whole bracketed elements within conditions being crossed if chromosomes have already been selected for crossover. E.g.
(stateof[0,0,1;mass] > 6) & (stateof[0,0,2;mass] < 6) – bits in bold can be exchanged for similarly logic-separated elements.
Note: *Ideally* chcons + chconsi + chcond + chb = 1 (for even distribution of types of crossover)
Note: The StringCrossoverModel class will perform only one of the above types. If it is found that the selected type of crossover cannot be performed, the crossover procedure is completed without any changes.

`mate-pop x`
x Denotes the proportion of the population (exact number not guaranteed) that will undergo mating. The population number will never exceed pop-max. Individuals are chosen at random. Note: all subsequent operations on the mating population only refers to this sub-population.

`offspring x y z`
x (floating point number) denotes the proportion (number not guaranteed) of the sub-population that will produce offspring (set to 0 if no offspring should be produced). The y (integer) denotes how many offspring there should be from a single mating between two parents. The z value (floating point number) denotes the proportion (number not guaranteed) of the offspring that will have a unique genome (not guaranteed – parental chromosomes are picked randomly in DefaultGAExecutionModel). Unique genomes are defined amongst offspring rather than against parents.

`cswap x`
Proportion (number not guaranteed) of chromosomes that should be swapped *from each individual* in the mating process.

`cswap-strict x`
Same as cswap except exactly x (integer) number of chromosomes will be swapped *from each individual* in the mating process (Chromosomes are picked at random). If the number exceeds the number of

chromosomes in the individual, the mating process skips the current individual. Note: if the cswap-strict tag is present in the property file then the value for the cswap tag is ignored. Therefore there MUST be either one or the other in the file (cswap or cswap-strict, but not both).

`parental-death` x
Proportion (number not guaranteed) of parents that are deleted from the population after mating. In DefaultMatingModel parental death only occurs if offspring are made in the first place.

`ppool` x
(Population pool) The file (fully qualified location) to which chromosomal population is saved to (File separators should always be forward-slashes). This file is appended during the GA run so that chromosomal evolution can be tracked.

`mprob` x
Mutation rate, where *xxx* represents a floating-point number between 0 and 1. For example, if it was 0.6 – this means that on average 60% of the population will test positive for mutation in every cycle. Note: All other operations involving mutation only apply to this sub-population.

`pointm` x
Point mutation rate. Mutates table formats and raw numbers found in rules (if allowed, see next tag).

`atim` *true/false*
Allows table input mutations. If true, the part in the square brackets is mutated rather than the table format itself.

`deletion` x
Deletion rate. A part of the chromosome is deleted at random.

`slippage` x
Slippage rate. Finds viable *parts* of chromosomes/rules that can be replicated and within the chromosome itself. In StringRuleMutator a slippage can occur on a raw number (numeric values are repeated) – consequents that are separated by ";;" can also be done but remains unimplemented in the current version.

`shuffle` x
Shuffle rate. The order of the chromosome/rules is shuffled.

`crep` x
Whole-chromosomal/rule replication rate. A chromosome is picked at random and is replicated. Not yet implemented because MutatorModel does not have access to the individual that owns the chromosome.

`pop-start` x
Initial population number (*x* must be integer)

`pop-max` x
Maximum population number during run (*x* must be integer)

`pop-min` x
Minimum population number during run (*x* must be integer)

`generation-max` x
Maximum number of generations allowed in the GARun.

`initpopmut` `true/false`
If true, the execution model invokes the mutation model to mutate the initial population to ensure a diverse population.

`fsg` x
Fitness stopping function – if an individual fitness reaches this value the GA exits.

`indiv` x y
X, the optim.ga.Individual-implementing class (must have an empty constructor). Y, the semantic code for the individual in String format (e.g. 1.423.34)

`ctype` x
The optim.ga.Chromosome-implementing class (must have an empty constructor)

`mmics-init` x
x should represent a file path from which a Modelomics initialisation file is constructed. Note that the file content is copied and put into a true Modelomics init file. The file should contain all the normal details (see master-slave-config-file.doc) including initialisation of rules according to the individual in the current GA run.

`cgen` *Job ID*
The job labelled "cgen" is executed to generate a list of Chromosome (genome) objects. The objects must be stored in the ScriptExecutor with the variable ID "cgen". The object saved must be a java.util.List<Chromosome> object. By having this tag on a Job, the job has access to the GAExecutionModel (and therefore the other models) and so there is the freedom that one can use the GA models to create a novel genome every time the cgen-labelled job is called (can do this with the gaexe command).

`cgen-f` x
The xxx represents a file name where chromosomes can be read from. The StringChromoListMCT class is used to translate the contents of the list.

`cmct` x
MemorableContentTranslator-implementing class (must have empty contructor) used to translate chromosomes from file.

`ffmct`
MemorableContentTranslator-implementing class (must have empty contructor) used by the fitness function, which needs to read an output file (generated from the simulation) and compare against another output file or running application (generated by some mathematical model). In DefaultExecutionModel, the cmct and ffmct translators and jungled – when a chromosome needs to be read the cmct translator is used (i.e. is set to the Individual) and when the fitness functions needs to operate the ffmct is used. In other words, the translators are used dynamically as required and each Individual therefore has not dedicated translator.

`indout` x
The file to which individuals will output.

`modelrunner` *id* x y
The ModelRunner class (x) that the individual output is compared against. The ModelRunner class generically represents any mathematical/computational model and it is up to the ModelRunner-extending class to implement any code that is needed to query the model in question. The ModelRunner class is instantiated with a file object location (denoted by y – note that this is optional but if it is not needed for the particular implementation of ModelRunner it should be given a "null" value, and the empty constructor version of ModelRunner will be made) that either represents a data output file from a previously-run model or some representation of a model (e.g. an SBML or CellML file) – either way, it is up to the ModelRunner class to execute and return a query. Multiple modelrunner declarations are allowed in the property file but all must have a unique ID. Note: ModelRunners are instantiated as soon as they are read, not during a fitness function test or any other time during the GA run.

`modelrun-p id <t>x`
Sets parameters for a model runner with the given ID (if it exists). Internally, x is added to a list, which is passed to the ModelRunner every time it is queried. Therefore the order that these tags are important! It is also important to get the type of the parameter correct. Various "switches" are available to label a parameter to the correct type (*t*-). Types currently defined:

- <d> double
- <f> float
- <i> integer
- <l> (small L) long
- <s> string
- <b> Boolean (note that the following string should be a "true" or "false")

Note: the <> that occurs before and after t to denote a type. Also note that there is no space between the type declaration and the value.

The following tags are used to define the order in which basic GA processes are performed (note that the processes are performed cyclically until the maximum number of iterations is complete and/or a stopping condition is met):

```
mutate              x      y
crossover           x      y
mate                x      y
selection           x      y      mrID  a,b,c...
runj                x      jobID
runSim
cident-c            x
cident-i            x
cl-indout
writestate          x

gastarter           x
```
The started class that interprets the property file and generates models, etc., and the initial population (for string rules, it is sts.ga.StringGS).

`mmod x`
Mutation model definition, e.g. x should be the qualified class name of a MutationModel-implementing class.

`cmod x`
Crossover model definition, e.g. x should be the qualified class name of a CrossoverModel-implementing class.

`fmod x`
Fitness Function model definition, e.g. x should be the qualified class name of a FitnessFunctionModel-implementing class.

`matemod x`
Mating model definition, e.g. x should be the qualified class name of a MatingModel-implementing class.

`gamod x`
GA execution model definition, e.g. x should be the qualified class name of a GAExecutionModel-implementing class.

B4: Standard Master/Client Attributes

When MMaster and MSlave are started, they need IPs and SimControl objects to be passed to them. This information will be held in .prop files.

Reading Property Files

These property files are read by the modelomics.core.PropsReader class.

General consensus on these files is that they are tab-delimited for pair-associations, unless recognised otherwise by the appropriate parser. Also, any lines starting with "#" sign is ignored.

Jobs can be embedded into the init files by labelling as follows:

```
JOB id
...String lines for job...
ENDJOB
```

(NOTE the space, not tab, between the JOB and id fields)

Jobs are then executed in the main classes (Modelomics or ModelomicsWH) according to the id order in a run tag:

```
RUNJOBS
...list of IDs of jobs to be run in this order (+newline)...
ENDRUNJOBS
```

Jobs labelled with an ID that starts with "init" are executed first (i.e. the order they appear in the RUNJOBS tag, and are guaranteed to run before non-init jobs):

```
JOB init-id
...String lines for job...
ENDJOB
```

PropsReader does not do any running whatsoever! It just stores the data and the programmer must get these stored data elements (maps or lists) and use them appropriately.

IPs and hostnames are given by the host ID.

Master

File name: sim/master-init.prop
Needs:

- Server port information
- Client port information
- Client IP information
- SimControl information
- JivaContainerManager (JCM) information
- BufferListener-implementing classes
- Codebase information – can set this on command line as:

-Djava.rmi.server.codebase=file:/Users/manishpatel/modelomics/version/beta/modelomics.jar or whatever the path of the file is.

- Registry (RMI) port

e.g.:

```
server-port          x
client-port          x
portpair             x:y (server:client as integers)
codebase             file:/…/beta/modelomics.jar
#rmiregistry port number
reg-port             x
#the default jcm is modelomics.core.server.DefaultJCM
jcm                  x
simlog               x (where simlog will be situated)
#define whether SimControl should collate data
collate              true/false
#define whether SimControl should exit modelomics after
#collation
exitoncol            true/false
#define where collated files should be saved to (directory)
coldir               x
#BufferListener-implementing classes
blisten              x
#BufferListenerPool load property
bpool        x (integer)
#clients available for simulation
client-IP            x
client-IP            x
#or can enter IP range
client-IP-r          x.y.z.a:b (where a and b are integers and denote a range)
#or total number of clients expected
clientcount          x
#specify whether the output/ folder contents should be
#deleted on exit
dcoe                 true
#specify a clean System.exit(0) on the vm
vme                  true
#specify whether output files should be zipped
zipall               true
#specify all jobs
JOB id
…
ENDJOB
#run jobs in this order
RUNJOBS
…
ENDRUNJOBS
```

For the first version of SimControl, initialisation (of simulations) can be handled by specific Job IDs: initxxx, where *xxx* can be anything (so long as the ID starts with init).

Slave

File name: sim/slave-init.prop
Needs:

- Default Master IP
- Default self IP
- Listening port
- Sending port
- SimControl information
- BufferListener-implementing classes

- Codebase information – can set this on command line as:
-Djava.rmi.server.codebase=file:/Users/manishpatel/modelomics/version/beta/modelomics.jar or whatever the path of the file is.
- Registry (RMI) port
- RuleModel information

Listening and sending port information and SimControl information is almost same for this as is for MMaster (above).
e.g.:

```
server-port         x
client-port         x
portpair            x:y (server:client as integers)
#BufferListener-implementing classes
blisten             x
#BufferListenerPool load property
bpool               x (integer)
#rmiregistry port number
reg-port            x
simlog              output/simlog (where simlog will be situated)
#self ip
host                x  (local machine IP)
master-ip           xxxx
codebase            file:/…/beta/modelomics.jar
#specify whether the output/ folder contents should be
#deleted on exit
dooe                true
#specify a clean System.exit(0) on the vm
vme                 true
#specify how often states of StateContainers are written
global-write        x
stopping-index      x
#Default RuleModel
#NOTE that these can be changed on an individual basis in
#the scripts!
drm                 sts.rbs.DefaultRuleModel
#Default RuleBasedSystemControl class
rbsc                modelomics.core.sim.DefaultRBSControl
#set max model size for RuleBasedSystemControl (OPTIONAL)
rbsc-load           100
```

B5: Benchmarks

Node	MFlops
Node 1	27.53
Node 2	27.53
Node 3	27.5
Node 4	27.53
Node 5	27.55
Node 6	27.5
Node 7	27.5
Node 8	27.49
Node 9	27.44

Node 10	27.36
Node 11	27.41
Node 12	27.41
Node 13	27.37
Node 14	27.31
Node 15	27.31
Node 16	27.22

Table 1. Average MFlops out of 100 repeats for each node running the flops program as described in §5.3.1

Node 1 to	Time (ms)
Node 2	8.333333333
Node 3	8.509803922
Node 4	8.294117647
Node 5	8.215686275
Node 6	7.980392157
Node 7	8.235294118
Node 8	8.098039216
Node 9	8.156862745
Node 10	8.333333333
Node 11	8.392156863
Node 12	8.764705882
Node 13	8.884313725
Node 14	8.764705882
Node 15	9.215686275
Node 16	9.470588235

Table 2. Average times for each node to complete `testcomm` (§5.3.1).

Lattice Sites	**0 Slave**	**1 Slave**	**4 Slaves**	**8 Slaves**
125	5.013	5.015	5.009	5.119
216	5.068	5.007	5.039	5.043
343	10.031	10.022	9.09	8.579
512	5.003	10.023	9.589	9.183
729	10.003	10.0145	10.048	9.883
1000	10.017	15.0285	15.0485	14.811
3375	25.016	31.0695	25.468	24.908
8000	55.072	171.2805	156.176	96.255
15625	105.03	185.984	173.226	109.452
27000	186.547	232.543	205.893	146.118
64000	959.739	373.72	283.187	225.648
125000	4309.863	572.069	410.47	334.04

Table 3. Average times to complete standard diffusion model for varying number of slaves.

APPENDIX C:
Simulation-related Results

C1: Initialisation rules and GA results by diffusion model decomposition

Attributes			Result	Attributes			Result	Attributes			Result
Cr_p	M_p	Mt_p		Cr_p	M_p	Mt_p		Cr_p	M_p	Mt_p	
0	0	0	Not Tested	0.25	0.75	0.5	+	0.75	0.5	0	-
0	0	0.25	Not Tested	0.25	0.75	0.75	-	0.75	0.5	0.25	-
0	0	0.5	Not Tested	0.25	0.75	1	-	0.75	0.5	0.5	-
0	0	0.75	Not Tested	0.25	1	0	-	0.75	0.5	0.75	-
0	0	1	Not Tested	0.25	1	0.25	-	0.75	0.5	1	+
0	0.25	0	Not Tested	0.25	1	0.5	-	0.75	0.75	0	-
0	0.25	0.25	-	0.25	1	0.75	+	0.75	0.75	0.25	-
0	0.25	0.5	-	0.25	1	1	-	0.75	0.75	0.5	-
0	0.25	0.75	-	0.5	0	0	Not Tested	0.75	0.75	0.75	-
0	0.25	1	-	0.5	0	0.25	-	0.75	0.75	1	+
0	0.5	0	Not Tested	0.5	0	0.5	-	0.75	1	0	-
0	0.5	0.25	-	0.5	0	0.75	-	0.75	1	0.25	-
0	0.5	0.5	-	0.5	0	1	-	0.75	1	0.5	-
0	0.5	0.75	-	0.5	0.25	0	-	0.75	1	0.75	+
0	0.5	1	-	0.5	0.25	0.25	-	0.75	1	1	-
0	0.75	0	Not Tested	0.5	0.25	0.5	-	1	0	0	Not Tested
0	0.75	0.25	-	0.5	0.25	0.75	+	1	0	0.25	-
0	0.75	0.5	-	0.5	0.25	1	+	1	0	0.5	-
0	0.75	0.75	-	0.5	0.5	0	-	1	0	0.75	-
0	0.75	1	-	0.5	0.5	0.25	-	1	0	1	-
0	1	0	Not Tested	0.5	0.5	0.5	-	1	0.25	0	-
0	1	0.25	-	0.5	0.5	0.75	-	1	0.25	0.25	-
0	1	0.5	-	0.5	0.5	1	+	1	0.25	0.5	-
0	1	0.75	-	0.5	0.75	0	-	1	0.25	0.75	-
0	1	1	-	0.5	0.75	0.25	-	1	0.25	1	-
0.25	0	0	Not Tested	0.5	0.75	0.5	-	1	0.5	0	-
0.25	0	0.25	-	0.5	0.75	0.75	-	1	0.5	0.25	-
0.25	0	0.5	-	0.5	0.75	1	-	1	0.5	0.5	-
0.25	0	0.75	-	0.5	1	0	-	1	0.5	0.75	-
0.25	0	1	-	0.5	1	0.25	-	1	0.5	1	+
0.25	0.25	0	-	0.5	1	0.5	-	1	0.75	0	-
0.25	0.25	0.25	-	0.5	1	0.75	+	1	0.75	0.25	-
0.25	0.25	0.5	-	0.5	1	1	-	1	0.75	0.5	-
0.25	0.25	0.75	-	0.75	0	0	Not Tested	1	0.75	0.75	+
0.25	0.5	0.25	-	0.75	0	0.25	-	1	0.75	1	-
0.25	0.5	0.5	+	0.75	0	0.5	-	1	1	0	-
0.25	0.5	0.75	+	0.75	0	0.75	-	1	1	0.25	-
0.25	0.5	1	-	0.75	0	1	-	1	1	0.5	-
0.25	0.75	0	-	0.75	0.25	0	-	1	1	0.75	-
0.25	0.25	1	+	0.75	0.25	0.25	-	1	1	1	-
0.25	0.5	0	-	0.75	0.25	0.5	+				

Table 1. Combination of attributes that achieved at least one set of chromosomes that performed at 95% accuracy when compared to the PDE solution of the diffusion equation (indicated by +) and populations in which no solution could be found (indicated by -, in which case the number of individuals either remained steady with no improvement in performance, or the population dwindled to 0). With the aim of finding a combination of attributes that could sustain a diverse population to determine the most favourable attributes for producing positive results only the three main genetic operators were tested with the diffusion model. These operators are: Crossover Probability (with sub-attributes remaining equally distributed), Mutation Probability and Mating Probability (leaving nOff=2, uGen=1, pD=1), labelled above as Cr_p, M_p, Mt_p respectively. All experiments were initialised with a population of 100 individuals with a single chromosome each (§??). The fitness function cut-off attribute was kept at 75%, i.e. the bottom 25% performers were eliminated from the population. Note that not all attribute combinations were tested (notably those in which at least 2 attributes were set to 0 probability). Additionally not all experiments could be replicated due to the time scale at which a single population might take to produce viable individuals – this was noted to be both unpredictable and not always reproducible. Section B3 details the various attributes that can be used to fine-tune a GA experiment. To test all of these was impractical, considering the number of different models that had to be decomposed and the time it would take to perform simulations with all permutations (especially where the setup would lead to hanging populations).

BIBLIOGRAPHY

Adam, JA., and N. Bellomo (1996), A Survey of Models for Tumor-Immune System Dynamics. Boston, USA: Birkhauser.

Adiwijaya, BS., PI. Barton, and B. Tidor (2006), "Biological network design strategies: discovery through dynamic optimization." Molecular BioSystems 2:650-659.

Alarcon, T., HM. Byrne, and PK. Maini (2004), "Towards whole-organ modelling of tumour growth." Progress in Biophysics and Molecular Biology 85:451-472.

Alber, Mark, Maria Kiskowski, James Glazier, and Yi Jiang (2002), "On Cellular Automaton Approaches to Modeling Biological Cells", Journal of Applied Mathematics, IMA 134 (Mathematical systems theory in biology, communication and finance):12.

Alberghina, L., and AM. Colangelo (2006), "The modular systems biology approach to investigate the control of apoptosis in Alzheimer's disease neurodegeneration." BMC Neuroscience 7 (Suppl 1):S2.

Ammar, EE., and ML. Hussein (2003), ""Radiotherapy problem under fuzzy theoretic approach." Chaos, Solitons and Fractals 18:739-744.

Anderson, AR. (2003), "A Hybrid Discrete-continuum Technique for Individual-based Migration Models", in W. Alt, MA. Chaplain, M. Griebel and K. Lenz (eds.), Polymer and Cell Dynamics - Multiscale Modeling and Numerical Simulations, Basel, Switzerland: Birkhauser Verlag.

Anderson, ARA., and MAJ. Chaplain (1998), "Continuous and Discrete Mathematical Models of Tumor-induced Angiogenesis." Bulletin of Mathematical Biology 60:857-900.

Antipas, V.P., G. S. Stamatakos, N. K. Uzunoglu, D.D. Dionysiou, and R.G. Dale (2004), "A spatio-temporal simulation model of the response of solid tumours to radiotherapy in vivo: parametric validation concerning oxygen enhancement ratio and cell cycle duration", Physics in Medicine and Biology 49:1485-1504.

Arakelyan, L., V. Vainstein, and Z. Agur (2002), "A computer algorithm describing the process of vessel formation and maturation, and its use for predicting the effects of anti-angiogenic and anti-maturation therapy on vascular tumor growth." Angiogenesis 5:203-214.

Araujo, RP., and DLS. McElwain (2004), "A History of the Study of Solid Tumour Growth: The Contribution of Mathematical Modelling", Bulletin of Mathematical Biology 66:1039-1091.

Arciero, JC., TL. Jackson, and DE. Kirschner (2004), "A Mathematical Model of Tumour-Immune Evasion and siRNA Treatment." Discrete and Continuous Dynamical Systems Series B 4 (1):39-58.

Arnaud, CH. (2006), "Systems Biology's Clinical Future", Chemical & Engineering News 84 (31):17-26.

Arndt, O., T. Barth, B. Freisleben, and M. Grauer (2005), "Approximating a finite element model by neural network prediction for facility optimization in groundwater engineering." European Journal of Operations Research 166:769-781.

Awale, S., J. Lu, SK. Kalauni, Y. Kurashima, Y. Tezuka, S. Kadota, and H. Esumi (2006), "Identification of Arctigenin as an Antitumor Agent Having the Ability to Eliminate the Tolerance of Cancer Cells to Nutrient Starvation", Cancer Research 66:1751-1757.

Ayton, Gary, and Gregory Voth (2002), "Bridging Microscopic and Mesoscopic Simulations of Lipid Bilayers", Biophysical Journal 83:3357-3370.

Back, AP, and TP Chen (1997), "Approximation of hybrid systems by neural networks", Paper read at Proc of 1997 Int Conf on Neural Information Processing.

Bagrodia, R., R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and HY. Song (1998), "Parsec: A Parallel Simulation Environment for Complex Systems." IEEE Computer 31 (10):77-85.

Baish, J., and RK. Jain (2000), "Fractals and Cancer." Cancer Research 60:3683-3688.

Baldi, P., and AD. Long (2001), "A Bayesian framework for the analysis of microarray expression data: regularised t-test and statistical inferences of gene changes." Bioinformatics 17 (6):509-519.

Bar-Yam, Y. (1997), Dynamics of Complex Systems: Studies in Non-Linearity. Oxford, UK: Perseus Press.

Bar-Yam, Y. (2004), "A Mathematical Theory of Strong Emergence Using Multiscale Variety", Complexity 9 (6):15-24.

Barabasi, AL., and ZL. Oltvai (2004), "Network Biology: Understanding the Cell's Functional Organization", Nature 5:101-113.

Baral, C., K. Chancellor, N. Tran., NL. Tran, A. Joy, and M. Berens (2004), "A knowledge based approach for representing and reasoning about signalling networks." Bioinformatics 20 (1):i15-i22.

Barton, Russell (1998), "Simulation Metamodels", Paper read at Proceedings of the 1998 Winter Simulations Conference.

Bates, RC., NS. Edwards, and JD. Yates (2000), "Spheroids and cell survival." Critical Reviews in Oncology Hematology 36:61-44.

Bedau, MA. (1997), "Weak Emergence", in J. Tomberlin (ed.), Philosophical Perspectives: Mind, Causation, and World, Malden, MA: Blackwell, 375-399.

Beeken, WD., W. Kramer, and D. Jonas (2000), "New molecular mediators in tumor angiogenesis." Journal of Cellular and Molecular Medicine 4 (4):264-269.

Bernier, Francois, Denis Poussart, Denis Laurendeau, and Martin Simoneau (2002), "Interaction-Centric Modelling for

Interactive Virtual Worlds: The APIA Approach", Paper read at 16th International Conference on Pattern Recognition (ICPR'02).

Bertholet, RG, DC Brogan, PF Reynolds, and JC Carnaham (2004), "In Search of the Philosopher's Stone: Simulation Composability Versus Component-Based Software Design", Paper read at Proceedings of the 2004 Fall Simulation Interoperability Workshop, at Orlando, Florida, USA.

Bertuzzi, A., and A. Gandolfi (2000), "Cell Kinetics in a Tumor Cord." Journal of Theoretical Biology 204:587-599.

Bhattacharya, SS., C. Brooks, E. Cheong, J. Davis, M. Goel, B. Kienhuis, AL. Edward, J. Liu, X. Liu, L. Muliadi, S. Neuendorffer, J. Reekie, N. Smyth, J. Tsay, B. Vogel, W. Williams, Y. Xiong, Y. Zhao, and H. Zheng (2005), "Ptolemy II: Heterogeneous Concurrent Modeling and Design in Java", in C. Brooks, AL. Lee, X. Liu, S. Neuendorffer, Y. Zhao and H. Zheng (eds.): University of Berkeley.

Bloomberg, J. (2004), "The SOA Implementation Framework - The Future of Service-Oriented Architecture and Software", ZapThink, LLC.

Boogerd, FC., FJ. Bruggerman, RC. Richardson, A. Stephan, and HV. Westerhoff (2005), "Emergence and its Place in Nature: A Case Study of Biochemical Networks", Synthese 145:131-164.

Borkenstein, K., S. Levegrun, and P. Paschke (2004), "Modeling and Computer Simulations of Tumor Growth and Tumor Response to Radiotherapy." Radiation Research 162:71-83.

Bossomaier, T., T. Cranny, and D. Schneider (1999), "A New Paradigm for Evolving Cellular Automata Rules", Paper read at Evolutionary Computation, at Washington DC, USA.

Bourgine, P., and J. Johnson (2006), "Living Roadmap for Complex Systems Science", in, European Conference on Complex Systems (ECCS06), Oxford, UK.

Brazma, Alvis, Pascal Hingamp, John Quackenbush, Gavin Sherlock, Paul Spellman, Chris Stoeckert, John Aach, Wilhelm Ansorge, Catherine A. Ball, Helen C. Causton, Terry Gaasterland, Patrick Glenisson, Frank C. P. Holstege, Irene F. Kim, Victor Markowitz, John C. Matese, Helen Parkinson, Alan Robinson, Ugis Sarkans, Steffen Schulze-Kremer, Jason Stewart, Ronald Taylor, Jaak Vilo, and Martin Vingron (2001), "Minimum information about a microarray experiment (MIAME)[mdash]toward standards for microarray data", Nat Genet 29 (4):365-371.

Bruggerman, FJ., and HV. Westerhoff (in press), "The nature of systems biology." Trends in Microbiology.

Bulashevska, S., O. Szakacs, B. Brors, R. Eils, and G. Kovacs (2004), "Pathways of urothelial cancer progression suggested by Bayesian network analysis of allelotyping data." International Journal of Cancer 110 (6):850-856.

Byrne, HM. () (1999), "Using mathematics to study solid tumour growth", Proceedings of the 9th General Meeting of European Women in Mathematics:81-107.

Carlson, JM., and J. Doyle (2002), "Complexity and Robustness", Proceedings in the National Academy of Science USA 99 (Suppl 1):2538-2545.

Catto, J. W., D. A. Linkens, M. F. Abbod, M. Chen, J. L. Burton, K. M. Feeley, and F. C. Hamdy (2003), "Artificial intelligence in predicting bladder cancer outcome: a comparison of neuro-fuzzy modeling and artificial neural networks", Clin Cancer Res 9 (11):4172-4177.

Chalmers, D. (2002), "Conciousness and its Place in Nature", in S. Stich and F. Warfield (eds.), Guide to the Philosophy of Mind., Oxford: Blackwell.

Channabasavaiah, K., and K. Holley (2004), "Migrating to service-oriented architectures." IBM On demand operating environment solutions.

Chaplain, MA., SR. McDougall, and AR. Anderson (2006), "Mathematical Modeling of Tumor-Induced Angiogenesis", The Annual Review of Biomedical Engineering 8:233-257.

Chari, Kaushal, and Tarun K Sen (1998), "An implementation of a graph-based modeling system for structured modeling (GBMS SM)", Decision Support Systems 22:103-120.

Charniak, E. (1991), "Bayesian Networks without Tears." AI Magazine 12 (4):64-91.

Charusanti, P., X. Hu, L. Chen, D. Neuhauser, and JJ. DiStefano III (2004), "A Mathematical Model of BCR-ABL Autophosphorylation, Signaling Through the CRKL Pathway, and Gleevec Dynamics in Chronic Myeloid Leukemia." Discrete and Continuous Dynamical Systems Series B 4 (1):99-114.

Chen, CC, S. Nagl, and CD Clack (submitted), "Computational Techniques for Modeling and Simulating Biological Systems", ACM Computing Surveys.

Chen, S., S. Ganguli, and CA. Hunt (2004), "An Agent-based Computational Approach for Representing Aspects of In Vitro Multi-cellular Tumor Spheroid Growth", Paper read at Proceedings of the 26th Annual International Conference of the IEEE EMBS, at San Francisco.

Chignola, R., A. Schenetti, G. Andrighetto, E. Chiesa, R. Foroni, S. Sartoris, G. Tridente, and D. Liberati (2000), "Forecasting the growth of multicell tumour spheroids: implications for the dynamic growth of solid tumours." Cell Proliferation 33:219-229.

Chin, DC., and AC Biondo (2002), "Multiple Neural Network Model Interpolation", Paper read at PerMIS Workshop, at Gaithersburg, MD, USA.

Cho, KH., and O. Wolkenhauer (2003), "Analysis and modelling of signal transduction pathways in systems biology." Biochemical Society Transactions 31 (6):1503-1509.

Christopher, R., A. Dhiman, J. Fox, R. Gendelman, T. Haberitcher, D. Kagle, G. Spizz, IG. Khalil, and C. Hill (2004), "Data-Driven Computer Simulation of Human Cancer Cell." Annals in the New York Academy of Sciences 1020:132-153.

Coffey, DS. (1998), "Self-organisation, complexity and chaos: The new biology for medicine." Nature Medicine 4 (8):882-885.

Colonna, A, V. Di Stefano, S. Lombardo, L. Papini, and GA. Rabino (2006), Learning Cellular Automata: Modelling

Urban Modelling 1998 [cited 12 12 2006]. Available from http: www.geocomputation.org 1998 89 gc 89.htm.

Cornish-Bowden, A. (2005), "Making systems biology work in the 21st century." Genome Biology 6:317-318.

Cornish-Bowden, A. (2006), "Putting the Systems back into Systems Biology", Perspectives in Biology and Medicine 49:475-489.

Coveney, Peter V, and Philip W Fowler (2005), "Modelling biological complexity: a physical scientists perspective." Journal of the Royal Society Interface 2 (4):267-280.

Covitz, PA., F. Hartel, C. Schaefer, SD. Coronado, G. Fragoso, H. Sahni, S. Gustafson, and KH. Buetow (2003), "caCORE: A common infrastructure for cancer informatics." Bioinformatics 19 (18):2404-2412.

Csete, M., and JC. Doyle (2002), "Reverse Engineering of Biological Complexity." Science 295:1664-1669.

D'Onofrio, A., and A. Gandolfi (2004), "Tumour eradication by antiangiogenic therapy: analysis and extensions of the model by Hahnfeldt et al. (1999)." Mathematical Biosciences 191 (2):159-184.

Dasu, A., I. Toma-Dasu, and M. Karlsson (2003), "Theoretical simulation of tumour oxygenation and results from acute and chronic hypoxia." Physics in Medicine and Biology 48:2829-2842.

Davies, PCW. (2004), "Emergent Biological Principles and the Computational Properties of the Universe", Complexity 10 (2):11-15.

Day, R., W. Shirey, S. Ramakrishnan, and Q. Huang (1998), "Tumour biology modeling workbench for prospectively evaluating cancer treatments." Paper read at Proceedings of the IEEE Computational Engineering with Systems Applications, at Hammamet, Tunisa.

de Castro, LN, and J. Timmis (2002), "Aritifical Immune Systems: A Novel Paradigm to Pattern Recognition", Paper read at Artificial Neural Networks in Pattern Recognition, at Paisley, UK.

De Pillis, LG., and A. Radunskaya (2001), "A Mathematical Model with Immune Resistance and Drug Therapy: an Optimal Theory Approach." Journal of Theoretical Medicine 3:79-100.

De Pillis, LG., and A. Radunskaya (2003), "The Dynamics of an Optimally Controlled Tumor Model: A Case Study." Mathematical and Computer Modelling 37:1221-1244.

De Pillis, LG., A. Radunskaya, and CL Wiseman (2005), "A Validated Mathematical Model of Cell-Mediated Immune Response to Tumor Growth", Cancer Research 65 (17):7950-7958.

Deisboeck, TS., ME. Berens, AR. Kansal, S. Torquato, AO. Stemmer-Rachamimov, and EA. Chiocca (2001), "Pattern of self-organisation in tumour systems: complex growth dynamics in a novel brain tumour spheroid." Cell Proliferation 34:115-134.

Delden, HV, P Luja, and G Engelen (in press), "Integration of multi-scale dynamic spatial models of socio-economic and physical processes for river basin management", Environmental Modelling & Software:1-16.

Desoize, B. (2000), "Contribution of three-dimensional culture to cancer research", Critical Reviews in Oncology Hematology 36:59-60.

Dhar, P., TC. Meng, S. Somani, L. Ye, K. Sakharkar, A. Krishnan, ABM. Ridwan, SHK. Wah, M. Chitre, and Z. Hao (2005), "Grid Cellware: the first grid-enabled tool for modelling and simulating cellular processes." Bioinformatics 21 (7):1284-1287.

Dionysiou, D.D., and G. S. Stamatakos (2006), "Applying a 4D multiscale in vivo tumor growth model to the exploration of radiotherapy scheduling: The effects of weekend treatment gaps and p53 gene status on the response of fast growing solid tumors", Cancer Informatics 2:113-121.

Dokoumetzidis, A., V. Karalis, A. Iliadis, and P. Macheras (2004), "The heterogeneous course of drug transit through the body." Trends in Pharmacological Sciences 25 (3):140-146.

Dolk, DR. (1993), "An introduction to model integration and integrated modeling environments." Decision Support Systems 10:249-254.

Dolk, D. (2000), "Integrated model management in the data warehouse era", European Journal of Operations Research 122:199-218.

Dolk, D. , and J. Kottemann (1993), "Model Integration and a Theory of Models", Decision Support Systems 9 (1):51-63.

Dormann, S., and A. Deutsch (2002), "Modeling of self-organized avascular tumor growth with a hybrid cellular automaton." In Silico Biology 2:0035.

Drasdo, D., and S. Hohme (2005), "A single-cell-based model of tumor growth in vitro: monolayers and spheroids", Physical Biology 2:133-147.

Drew, N., and R. Hendley (1995), "Visualising Complex Interaction Systems." Companion to ACM Conf. Human Factors in Computing Systems:204-205.

Dubitzky, W. (2006), "Understanding the computational methodologies of systems biology", Briefings in Bioinformatics 7 (4):315-317.

Duchting, W., and Y. Vogelsanger (1981), "Three-dimensional pattern generation applied to spheroidal tumor growth in a nutrient medium." International Journal of Biomedical Computing 12:377-392.

Eddy, SR. (2004), "What is Bayesian statistics?" Nature Biotechnology 22 (9):1177-1178.

Ferriera, SC., ML Martins, and MJ Vilela (2002), "Reaction-diffusion model for the growth of avascular tumor", Physics Reviews E (65):051914.

Finkelstein, A., J. Hetherington, L. Li, O. Margoninski, P. Saffrey, R. Seymour, and A. Warner (2004), "Computational Challenges of Systems Biology." IEEE Computer 37 (4):26-33.

Fisher, MJ., G. Malcolm, and RC. Paton (2000), "Spatio-logical processes in intracellular signalling." BioSystems 55:83-92.

Fitzpatrick, R. Euler's Method. 2003 [cited. Available from http: farside.ph.utexas.edu teaching 329 lectures node60.html].

Flynn, AA., AJ. Green, BR. Pedley, GM. Boxer, J. Dearling, R. Watson, R. Boden, and RH. Begent (2002), "A model-based approach for the optimisation of radioimmunotherapy through antibody design and radionulide selection." Cancer Research 15 (94):1249-1257.

Frantz, Frederick (1995), "A taxonomy of model abstraction techniques", Paper read at Proceedings of the 1995 Winter SImulation Conference.

Freyer, J., Y. Jiang, and Pjesivac. A Cellular Model for Avascular Tumor Growth. [cited. Available from www.ki.se icsb2002 pdf ICSB_123-124.pdf].

Fricks, R., C. Hirel, S. Wells, and KS. Trivedi (1997), "The Development of an Integrated Modeling Environment", Paper read at Proceedings of the 1st World Conference on Systems Simulation, WCCS.

Friedman, A. (2004), "A hierarchy of cancer models and their mathematical challenges", Continuous and Discrete Dynamical Systems - Series B 4 (1):147-159.

Friedman, N., M. Linial, I. Nachman, and D. Pe'er (2000), "Using Bayesian networks to analyze expression data." Journal of Computational Biology 7 (3-4):601-620.

Frigyesi, A., D. Gisselsson, F. Mitelman, and M. Hoglund (2003), "Power Law Distribution of Chromosome Aberrations in Cancer." Cancer Research 63:7094-7097.

Gatenby, RA., and PK. Maini (2003), "Mathematical oncology: Cancer summed up", Nature 421: 321-324.

Gaudet, S. "A computational model of the TNF signaling network can classify cells based on their kinetic caspase activation."

Gavaghan, S. Integrative Biology Group [cited. Available from http: web.comlab.ox.ac.uk oucl research areas biocomp tumour.html].

Gazit, Y., JW. Baish, N. Safabakshsh, M. Leunig, LT. Baxter, and RK. Jain (1997), "Fractal Characteristics of Tumor Vascular Architecture During Tumor Growth and Regression." Microcirculation 4 (4):395-402.

Geoffrion, AM. (1987), "An introduction to structured modeling", Management Science 33:547-588.

Geoffrion, AM. (1989a), "The formal aspects of structured modeling", Operations Research 37 (1):30-51.

Geoffrion, AM. (1989b), "Integrated modeling systems", Computer Science in Economics and Management 2 (3-15).

Geoffrion, AM. (1999), "Structured Modeling: Survey and Future Research Directions." ORSA CSTS Newsletter 15:1.

Giavitto, JL., and C. Godin Computational Models for Integrative and Developmental Biology 2002 [cited. Available from http: mgs.lami.univ-evry.fr PUBLICATIONS lami-RR72--computational-models-for-dynamical-structure.pdf].

Giraldi, GA, LC da Costa, AV Xavier, and PS Rodrigues (2005), "Methods for Analytical Understanding of Agent-Based Modeling of Complex Systems", ArXiv Computer Science e-prints.

Glass, L. (1973), "Instability and mitotic patterns in tissue growth." Journal of Dynamic Systems and Measurement Control 95:324-327.

Gonzalez, PP. 2002 [cited. Available from http: www-dii.ing.unisi.it aiia2002 paper BIOINFO gonzalez-aiia02.pdf].

Greenspan, HP. (1972), "Models for the growth of solid tumour by diffusion", Studies in Applied Mathematics 52:317-340.

Hakman, M., and T. Groth (1999), "Object-oriented biomedical system modelling - the language." Computer Methods and Programs in Biomedicine 60:153-181.

Hall, P. Cancer Basics - Growth Curve 2003 [cited. Available from http: www.qub.ac.uk cm pat undergraduate Basiccancer growth_curve.htm].

Harold, ER. (2004), Java Network Programming: O'Reilly.

Harris, AL. (1997), "Antiangiogenesis for cancer therapy." Lancet 349 (2):13-15.

Hauser, H. 1999 [cited. Available from http: www.cg.tuwien.ac.at research vis dynsys].

Hetherington, J., A. Warner, and R. Seymour (2006), "Simplification and its consequences in biological modelling: conclusions from a study of calcium oscillations in hepatocytes", Journal of the Royal Society Interface 3:319-331.

Hill, AV. (1928), "The diffusion of oxygen and lactic acid through tissues", Proceedings of the Royal Society of London B, Biological Sciences 104:39-96.

Hnat, B., and SC. Chapman (2000), "Visualisation of multiscale datasets in numerical models of complex systems." International Journal of Computer Integrated Design and Construction 2 (4):235-240.

Holash, J., SJ. Wiegandand, and GD. Yancopoulos (1999), "New model of tumor angiogenesis: Dynamic balance between vessel regression and growth mediated by angiopoietins and VEGF", Oncogene 18 (38):5356-5362.

Holden, C. (2002), "Alliance Launched to Model E.coli", Science 297:1459-1460.

Holland, JH (1975), Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press.

Hu, YF., RJ. Allan, and KCF. Maguire (2006), Comparing the performance of JAVA with Fortran and C for numerical computing 2000 [cited 2006]. Available from http: www.dl.ac.uk TCSC UKHEC JASPA bench.html.

Hucka, M., A. Finney, HM. Sauro, H. Bolouri, JC. Doyle, H. Kitano, AP. Arkin, BJ. Bornstein, D. Bray, A. Cornish-Bowden, AA. Cuellar, S. Dronov, ED. Gilles, M. Ginkel, V. Gor, II. Goryanin, WJ. Hedley, TC. Hodgman, JH. Hofmeyr, PJ. Hunter, NS. Juty, JL. Kasberger, A. Kremling, U. Kummer, N. Le Novere, LM. Loew, D. Lucio, P. Mendes, E. Minch, ED. Mjolsness, Y. Nakayama, MR. Nelson, PF. Nielsen, T. Sakurada, JC. Schaff, BE. Shapiro, TS. Shimizu, HD. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang (2003), "The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models", Bioinformatics 19 (4):524-531.

Hunter, PJ. (2004), "The UIPS Physiome Project: a framework for computational physiology." Progress in Biophysics and Molecular Biology 85:551-569.

Hunter, PJ., and TK. Borg (2003), "Integration from proteins to organs: the Physiome Project." Nature Reviews Molecular Cell Biology 4:237-243.

Hussein, ML., and E. Ahmed (2000), "Fuzzy concepts in radiotherapy." Fuzzy Sets and Systems 114:305-309.

Hyman, JM (2005), "Patch Dynamics for Multiscale Problems", Computing in Science and Engineering 7 (3):47-53.

Ideker, T., and D. Lauffenburger (2003), "Building with a scaffold: emerging strategies for high- to low-level cellular modeling", Trends in Biotechnology 21 (6):255-262.

Iskandarani, M, D. Haidvogel, JC. Levin, E. Curchitser, and CA. Edwards (2002), "Multiscale Geophysical Modeling using the Spectral Element Method", Computing in Science and Engineering 4 (5):42-48.

Jackson, TL., and HM. Byrne (2000), "A mathematical model to study the effects of drug resistance and vasculature on the response of solid tumors to chemotherapy." Mathematical Biosciences 164 (1):17-38.

Janes, AJ, JG Albeck, S. Gaudet, PK Sorger, D. Lauffenburger, and MB Yaffe (2005), "A Systems Model of Signaling Identifies a Molecular Basis Set for Cytokine-Induced Apoptosis", Science 310 (5754):1646-1653.

Jensen, FV. (2001), Bayesian Networks and Decision Graphs. Germany: Springer-Verlag.

Jiang, Y., J. Pjesivac-Grbovic, C. Cantrell, and J. Freyer (2005), "A Multiscale Model for Avascular Tumor Growth", Biophysical Journal 89:3884-3894.

Johnson, S. (2001), Emergence. USA: The Penguin Group.

Kanoh, H., and Y. Wu (2003), "Evolutionary Design of Rule Changing Cellular Automata", Paper read at International Conference on Knowledge-Based Intelligent Information and Engineering Systems.

Kansal, AR., S. Torquato, GR. Harsh IV, EA. Chiocca, and TS. Deisboeck (2000), "Simulated Brain Tumor Growth Dynamics Using a Three-Dimensional Cellular Automaton." Journal of Theoretical Biology 203:367-382.

Kasputis, S, and HC Ng (2000), "Composable Simulations", Paper read at Proceedings of the 2000 Winter Simulation Conference.

Kecman, V. (2001), Learning and Soft Computing. USA: MIT Press.

Kevrekidis, IG, CW Gear, JM Hyman, PG Kevrekidis, O Runborg, and C Theodoropoulos (2003), "Equation-Free Course Grained Multiscale Computation: Enabling Microscopic Simulators to Perform Systems-Level Tasks", Communications in the Mathematical Sciences 14:715-762.

Khu, ST, DA Savic, Y Liu, and H Madsen (2004), "A fast Evolutionary-based meta-modelling approach for the calibration of a rainfall-runoff model", Paper read at IEMSS conference, at Osnabruck.

Kitano, H. (2002), "Computational systems biology." Nature 420:206-210.

Kitano, H. (2003), "A graphical notation for biochemical networks." Biosilico 1 (5):169-176.

Kitano, H. (2004), "Biological Robustness", Nature Reviews Genetics 5:826-837.

Kohn, S., JA. Nagy, HF. Dvorak, and AM. Dvorak (1992), "Pathways of macromolecular tracer transport across venules and small veins. Structural basis for the hyperpermeability of tumor blood vessels." Laboratory Investigation 67:596-607.

Kosko, B. (1994), Fuzzy Thinking. USA: Flamingo.

Kosko, B., and S. Isaka Fuzzy Logic 1993 [cited. Available from
http: www.fortunecity.com emachines e11 86 fuzzylog.html].

Krishnan, Ramayya, and Kaushal Chari (2000), "Model Management: Survey, Future Directions and a Bibliography", Interactive Transactions of ORMS 3 (1).

Kunz-Schughart, LA., M. Kreutz, and R. Knuechel (1998), "Multicellular spheroids: a three-dimensional in vitro culture system to study tumour biology." International Journal of Experimental Pathology 79:1-23.

Kurata, Hiroyuki, Kouichi Masaki, Yoshiyuki Sumida, and Rei Iwasaka (2005), "CADLIVE dynamic simulator: Direct link of biochemical networks to dynamic models", Genome Research 15:590-600.

Lakmeche, A., and O. Arino (2001), "Nonlinear mathematical model of pulsed-therapy of heterogeneous tumors." Nonlinear Analysis: Real World Applications 2:455-465.

Le, Q., and GM. Knapp (2003), "Incorporation Fuzzy Logic Admission Control in Simulation Models." Paper read at Proceedings of the 2003 Winter Simulations Conference.

Lee, Dong-Yup, Choamun Yun, Ayoun Cho, Bo Kyeng Hou, Sunwon Park, and Sang Yup Lee (2006), "WebCell: a web-based environment for kinetic modeling and dynamic simulation of cellular networks", Bioinformatics 22 (9):1150-1151.

Lee, HK., AK. Hsu, J. Sajdak, J. Qin, and P. Pavlidis (2004), "Coexpression Analysis of Human Genes Across Many Microarray Data Sets." Genome Research 14:1085-1094.

Levine, HA., BD. Sleeman, and M. Nilsen-Hamilton (2001), "Mathematical modeling of the onset of capillary formation initiating angiogenesis", Journal of Mathematical Biology 42:195-238.

Li, Y., C. Campbell, and M. Tipping (2002), "Bayesian automatic relevance determination algorithms for classifying gene expression data." Bioinformatics 18 (10):1332-1339.

Liotta, LA., J. Kleinerman, and GM. Saidel (1974), "Quantitative relationships of intravascular tumor cells, tumor vessels, and pulmonary metastases following tumor implantation", Cancer Research 34:997-1004.

Lloyd, CM., MDB. Halstead, and PF. Nielsen (2004), "CellML: its future, present and past." Progress in Biophysics and Molecular Biology 85:433-450.

Loew, LM., and JC. Schaff (2001), "The Virtual Cell: a software environment for computational cell biology." Trends in Biotechnology 19 (10):401-406.

Lok, L. (2004), "The need for speed in stochastic simulation", Nature Biotechnology 22 (8):964-965.

Luigi, P. (2003), Cancer Modelling and Simulation. USA: Chapman & Hall CRC.

Ma, J., Q. Tian, and D. Zhou. (1998), "An Object-Oriented Approach to Structured Modeling." Paper read at IRMA '98 Proceedings1998 Information Resources Management Association International Conference.

Maisonpierre, PC., C. Suri, PF. Jones, S. Bartunkova, SJ. Wiegand, C. Radziejewski, D. Compton, J. McClain, TH. Aldrich, N. Papadopoulos, TJ. Daly, S. Davis, TN. Sato, and GD. Yancopoulos (1997), "Angiopoietin-2, a natural antagonist for Tie-2 that disrupts in vivo angiogenesis." Science 277:55-60.

Makowski, M (2005), "A structured modeling technology", European Journal of Operations Research 166 (3):615-648.

Maley, CC., and S. Forest (2000), "Exploring the relationship between neutral and selective mutations in cancer." Artificial Life 6 (4):325-345.

Mallet, DG, and LG. De Pillis (2006), "A cellular automata model of tumor-immune system interactions." Journal of Theoretical Biology 239:334-350.

Mangoine, C. (2006), Performance tests show java as fast as C++. Java World 1998 [cited 2006]. Available from http: www.javaworld.com jw-02-1998 jw-02-jperf.html.

Mansury, Y., and TS. Deisboek (2003), "The impact of 'search precision' in an agent-based tumor model." Journal of Theoretical Biology 224 (3):325-337.

Mantzaris, NV., S. Webb, and HG. Othmer (2004), "Mathematical modeling of tumor-induced angiogenesis." Journal of Mathematical Biology 49:111-187.

Margoninski, O., P. Saffrey, J. Hetherington, A. Finkelstein, and A. Warner (2006), "A Specification Language and a Framework for the Execution of Composite Models in Systems Biology", LNCS Transactions on Computational Systems Biology VII (LNBI 4230).

Marinov, M, and I Zheliazkova (2005), "An interactive tool based on priority semantic networks", Knowledge-Based Systems 18:71-77.

Markus, M., D. Bohm, and M. Schmick (1999), "Simulation of vessel morphogenesis using cellular automata", Mathematical Biosciences 156:191-206.

Marrow, P. (2000), "Nature-inspired computing technology and applications." British Telecom Technology Journal 18 (4):13-23.

Marusic, M., Z. Bajzer, S. Vuk-Pavlovic, and JP. Freyer (1994), "Tumor growth in vivo and as multicellular spheroids compared by mathematical models." Bulletin of Mathematical Biology 56 (4):617-631.

McDougall, SR., AR. Anderson, MA. Chaplain, and JA. Sherratt (2002), "Mathematical modelling of flow through vascular networks: implications for tumour-induced angiogenesis and chemotherapy strategies." Bulletin of Mathematical Biology 64 (4):673-702.

McElwain, DLS., and GJ. Pettet (1993), "Cell migration in multicell spheroids: swimming against the tide", Bulletin of Mathematical Biology 55:655-674.

Meng, TC., S. Somani, and P. Dhar (2004), "Modeling and simulation of biological systems with stochasticity", In Silico Biology 4:0024.

Miller, JH., and F. Zheng (2004), "Large Scale Simulations of Eukaryotic Cell Signaling Processes", Parallel Computing 30 (9-10):1137-1149.

Miller, JH., F. Zheng, S. Jin, LK. Opresko, HS. Wiley, and H. Resat (2005), "A model of cytokine shedding induced by low doses of gamma radiation." Radiation Research 163 (3):337-342.

Mitchell, M. (1996a), An Introduction to Genetic Algorithms. 2 ed. Boston, USA: MIT Press.

Mitchell, M., JP. Crutchfield, and R. Das (1996), "Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work", Paper read at EvCA'96.

Mizraji, E. (2004), "The Emergence of Dynamical Complexity: An Exploration Using Elementary Cellular Automata." Complexity 9 (6):33-42.

Moiola, JL. (1994), "An overview of bifurcation, chaos and nonlinear dynamics in control systems." Journal of the Franklin Institute 33 (6):819-858.

Mooney, D., and R. Swift (1999), A Course in Mathematical Modeling. USA: The Mathematical Association of America.

Moreira, JE., SP. Midfiff, PV. Gupta, M. Anrtigas, M Snir, and RD. Lawrence (2000), "Java Programming for high-performance numerical computing", IBM Systems Journal 39 (1):0-21.

Nagl, S., M. Williams, and J. Williamson (2007), "Objective Bayesian Nets for Systems Modelling and Prognosis in Breast Cancer", in D. Holms and LC. Jain (eds.), Innovations in Bayesian Networks: Theory and Applications: Springer.

Naistat, DM., and R. Leblanc (2004), "Proteomics", Journal of Environmental Pathology, Toxicology and Oncology 23 (3):161-178.

Natarajan, L., CC. Berry, and C. Gasche (2003), "Estimation of Spontaneous Mutation Rates." Biometrics 59:555-561.

Nelson, P (2003), Biological Physics: Energy, Information, Life: WH Freeman.

Netti, PA., LP. Baxter, R. Boucher, R. Skalak, and RK. Jain (1995), "Time-dependant behaviour of interstitial fluid pressure in solid tumors: implication for drug delivary." Cancer Research 55 (22):5451-5458.

Noble, D. (2005), "The heart is already working", Biochemical Society Transactions 33:539-542.

O'Connor, T., and HY. Wong (2002), Emergent Properties. Edited by EN Zalta, The Stanford Encyclopedia of Philosophy (Winter 2002 Edition).

O'Reilly, RC. (2006), "Biologically Based Computational Models of High-Level Cognition", Science 314 (5796):91-94.

Otter, M., and H. Elmqvist (2001), "Modelica: Language, Libraries, Tools, Workshop and EU-Project RealSim".

Papetti, M., and IM. Herman (2002), "Mechanisms of normal and tumor-derived angiogenesis." American Journal of Physiology - Cell Physiology 282:C947-C970.

Papin, JA., JL. Reed, and BO. Palsson (2004), "Unbiased modularization of biochemical networks: the example of correlated reaction sets." Trends in Biochemical Sciences 29:641-647.

Patel, AA., ET. Gawlinski, SK. Lemieux, and RA. Gatenby (2001), "A Cellular Automaton Model of Early Tumour Growth and Invasion: The Effects of Native Tissue Vascularity and Increased Anaerobic Tumour Metabolism", Journal of Theoretical Biology 213:315-331.

Patel, M., and S. Nagl (2004), "MicroCore: mapping genome expression to cell pathways and networks." Comparative and Functional Genomics 5:75-78.

Pawaletz, N., and M. Knierim (1989), "Tumor-related angiogenesis", Critical Reviews in Oncology and Hematology 9:197-242.

Pearl, J. (1988), Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. USA: Morgan Kaufmann (Elsevier).

Pienta, K. (2006), "Modeling cancer as a complex adaptive system: Genetic instability and evolution", in TS. Deisboeck and JY. Kresh (eds.), Complex Systems Science in Biomedicine.: Springer.

Pinho, STR., HI. Freedman, and F. Nani (2002), "A Chemotherapy Model for the Treatment of Cancer with Metastasis." Mathematical and Computer Modelling 36:773-803.

Plank, MJ., and BD. Sleeman (2003), "A reinforced random walk model of tumour angiogenesis and anti-angiogenic strategies." Mathematical Medicine and Biology 20 (2):135-181.

Please, CP., GJ. Pettet, and DLS. McElwain (1998), "A new approach to modelling the formation of necrotic regions in tumours." Applied Mathematics Letters 11:89-94.

Purwasih, W., and Y. Nakamori (2004), "Structured Modeling Technology: A Modeling Environment for Integrated Environmental Assessment", Complexity and Integrated Resources Management, Transactions of the 2nd Biennial Meeting of the International Environmental Modelling and Software Society iEMSs 2004.

Qi, AS., X. Zheng, CY. Du, and BS. An (1993), "A Cellular Automaton Model of Cancerous Growth." Journal of Theoretical Biology 161:1-12.

Raczynski, S. 1996 [cited.
Available from http: www.raczynski.com art cdssart2.pdf].

Ramil, JF., and MM. Lehman Fuzzy Dynamics in Software Project Simulation and Support [cited. Available from http: www.doc.ic.ac.uk ~mml feast2 papers pdf 595.pdf].

Reis, A.N.dos, J.C.M. Mombach, and M. Walter (2001), "The role of decreased cell adhesion in tumor morphology: A simulation study." Paper read at 4th International Eurosim 2001 Congress, at Delft, Netherlands.

Resnick, N., and MA. Gimbrone (1995), "Hemodynamic forces are complex regulators of endothelial gene expression", The FASEB Journal 9:874-882.

RIbba, B., T. Colin, and S. Schnell (2006), "A multiscale mathematical model of cancer, and it's use in analysing irradiation therapies." Theoretical Biology and Medical Modelling 3 (7):1-19.

Roxburgh, SH., and ID. Davies (2006), "COINS: an integrative modelling shell for carbon accounting and general ecological analysis", Environmental Modelling & Software 21:359-374.

Saffrey, Peter, Ofer Margoninski, James Hetherington, Marta Varela-Rey, Sachi Yamaji, Anthony Finkelstein, David Bogle, and Anne Warner (submitted), "End-to-end Information Management for Systems Biology ", Lecture Notes in Computer Science Transactions on Computational Systems Biology.

Santos, Isabel Reis dos, and AMO Porta Nova (1999), "The main issues in nonlinear simulation metamodel estimation", Paper read at Proceedings in the 1999 Winter Simulation Conference.

Sauro, HM., M. Hucka, A. Finney, C. Wellock, H. Bolouri, J. Doyle, and H. Kitano (2003), "Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration." OMICS 7 (4):355-372.

Scalerandi, M., B. Capogrosso Sansone, C. Benati, and CA. Condat (2002), "Competition effects in the dynamics of tumor cords." Physical Review E 65:051918.

Scalerandi, M, F. Peggion, B. Capogrosso Sansone, and C. Benati (2003), "Avascular and Vascular Phases in Tumor Cords Growth." Mathematical and Computer Modelling 37:1191-1200.

Scalerandi, M., and BC. Sansone (2002), "Inhibition of vascularisation in tumor growth." Physical Review Letters 89 (21):218101.

Schaefer, CF. (2004), "Pathway databases", Annals of the New York Academy of Sciences 1020:77-91.

Schoeberl, B., C. Eichler-Jonsson, ED. Gilles, and G. Muller (2002), "Computational modelling of the dynamics of the MAP kinase cascade activated by surface and internalised EGF receptors." Nature Biotechnology 20:370-375.

Sherrat, JA., and MAJ. Chaplain (2001), "A new mathematical model for avascular tumour growth." Journal of Mathematical Biology 43:291-312.

Shmulevich, I., E. R. Dougherty, S. Kim, and W. Zhang (2002), "Probabilistic Boolean Networks: a rule-based uncertainty model for gene regulatory networks", Bioinformatics 18 (2):261-274.

Siromoney, A., L. Raghuram, I. Korah, and GNS. Prasad (2000), "Inductive logic programming for knowledge discovery from MRI data." IEEE Engineering in Medicine and Biology Magazine 19 (2):72-77.

Slepchenko, BM., JC. Schaff, I. Macara, and LM. Loew (2003), "Quantitative cell biology with the Virtual Cell." Trends in Cell Biology 13 (11):570-576.

Smallbone, K., S. Gavaghan, RA. Gatenby, and P.K. Maini (2005), "The role of acidity in solid tumour growth and invasion", Journal of Theoretical Biology 235:476-484.

Smallwood, RH., and M. Holcombe (2007), The Epiltheliome Project: multiscale agent-based modeling of epithelial cells. 2006 [cited September 2007]. Available from http: www.dcs.shef.ac.uk ~rod Documents ISBI%202006% 20Smallwood.pdf.

Snoep, Jacky L, Frank Bruggerman, Brett G Olivier, and Hans V Westerhoff (2006), "Towards building the silicon cell: A modular approach", BioSystems 83 (2-3):207-216.

Sowa, JF Processes and Causality 2000 [cited. Available from http: www.jfsowa.com ontology causal.htm].

Spirtes, P., C. Glymour, R. Scheines, S. Kauffman, V. Aimale, and F. Wimberly (2000), "Constructing Bayesian Network models of Gene Expression Networks from Microarray Data." Paper read at Proceedings of the Atlantic Symposium on Computational Biology, Genome Information Systems & Technology.

Stamatakos, GS., NK. Uzunoglu, K. Delibasis, M. Makropoulou, N. Mouravliansky, and A. Marsh (1998), "A simplified simulation model and virtual reality visualization of tumour growth in vitro." Future Generation Computer Systems 14:79-89.

Stamatakos, GS., NK. Uzunoglu, K. Delibasis, M. Makropoulou, N. Mouravliansky, A. Marsh, EI. Zacharaki, and KS. Nikita (2001), "Modeling Tumor Growth and Irradiation Response in vitro - A combination of High-Performance Computing and Web-Based Technologies Including VRML Visualization." IEEE Transactions of Information Technology in Biomedicine 5 (4):279-289.

Stephanou, A., SR. McDougall, ARA. Anderson, MAJ. Chaplain, and JA. Sherratt (2005), "Mathematical Modelling of Flow in 2D and 3D Vascular Networks: Applications to Anti-angiogenic and Chemotherapeutic Drug Strategies", Mathematical and Computer Modelling 41:1137-1156.

Stewart, RD., and RJ. Traub (2000), "Temporal Optimization of Radiotherapy Treatment Fractions." Paper read at Proceedings of the ANS Topical Meeting on Radiation Protection for our National Priorities, Medicine, the Environment, and the Legacy (RPS 2000).

Stewart, RD., and RJ. Traub, "Radiobiological Modeling in Voxel Constructs 2000" [cited. Available from http: www.pnl.gov berc epub pnnl33487 mc2000x.pdf].

Stoll, BR., C. Migliorini, A. Kadambi, LL. Munn, and RK. Jain (2003), "A mathematical model of the contribution of endothelial progenitor cells to angiogenesis in tumors: implications for antiangiogenic therapy." Blood 102 (17):2555-2561.

Stott, EL., NF. Britton, JA. Glazier, and M. Zajac (1999), "Stochastic Simulation of Benign Avascular Tumour Growth Using the Potts Model." Mathematical and Computer Modelling 30:183-198.

Stromback, Lena, and Patrick Lambrix (2005), "Representations of molecular pathways: an evaluation of SBML, PSI MI and BioPAX." Bioinformatics 21 (24):4401-4407.

Succi, S., IV. Karlin, and H. Chen (2002), "Colloquium: Role of the H theorem in lattice Boltzmann hydrodynamic simulations." Reviews of Modern Physics 74:1203-1220.

Suh, CK., EH Suh, and DM Lee (1995), "Artificial itelligence approaches in model management systems: a survey", Computers and Engineering 28 (2):291-299.

Sung, MH., and R. Simon (2004), "In Silico Simulation of Inhibitor Drug Effects on Nuclear Factor-kappa B Pathway Dynamics." Molecular Pharmacology 66 (1):70-75.

Sutherland, RM., and RE. Durand (1971), "Growth of multicell spheroids in tissue culture as a model of nodular carcinomas." J. Natl. Cancer Inst. 46:113-120.

Takahashi, K., N. Ishikawa, Y. Sadamoto, H. Sasamoto, S. Ohta, A. Shiozawa, F. Miyoshi, Y. Naito, Y. Nakayama, and M. Tomita (2003), "E-Cell 2: Multi-platform E-Cell simulation system." Bioinformatics 19 (13):1727-1729.

Takahashi, K., K. Kaizu, B. Hu, and M. Tomita (2004), "A multi-algorithm, multi-timescale method for cell simulation." Bioinformatics 20 (4):538-546.

Takahashi, K., K. Yugi, K. Hashimoto, Y. Yamada, CJF. Pickett, and M. Tomita (2002), "Computational Challenges in Cell Simulation: A Software Engineering Approach." IEEE Intelligent Systems 17 (5):64-71.

Tan, WY., and CW. Chen (2000), "Assessing Effects of Changing Environment by a Multiple Pathway Model of Carcinogenesis." Mathematical and Computer Modelling 32:229-250.

Tay, Joc Cing, and Atul Jhavar (2005), "CASISS: A Complex Adaptive Framework for Immune System

Simulation", ACM Symposium on Applied Computing - Bioinformatics:158-164.

Thomas, D., and A. Hunt (2002), "State Machines", IEEE Software (November-December):10-12.

Tolman, HL., VM. Krasnopolky, and DV. Chalikov (2005), "Neural network approximations for nonlinear interactions in wind wave spectra: direct mapping for wind seas in deep water", Ocean Modelling 8:253-278.

Troyanskaya, OG., K. Dolinski, AB. Owen, RB. Altman, and D. Botstein (2003), "A Bayesian framework for combining heterogeneous data sources for gene prediction (in Saccharomyces cerevisiae)." PNAS 100 (14):8348-8353.

Tsai, YC. (2001), "Comparative analysis of model management and relational database management." Omega 29:157-170.

Tyson, JJ., K. Chen, and B. Novak (2001), "Network Dynamics and Cell Physiology." Nature Molecule Cell Biology 2:908-918.

Ubezio, P. (2004), "Unraveling the Complexity of Cell Cycle Effects of Anticancer Drugs in Cell Populations." Discrete and Continuous Dynamical Systems Series B 4 (1):323-335.

Uhrmacher, AM., D. Degenring, and B. Zeigler (2005), "Discrete Event Multi-level Models for Systems Biology", in, Transactions on Computational Systems Biology I, Berlin Heidelberg: Springer.

Van Dyke Parunak, H (1999), "Industrial and Practical Applications of DAI", in G Weiss (ed.), Multiagent Systems: a modern approach to distributed artificial intelligence: MIT Press, 643.

Vangheluwe, Hans, Juan de Lara, and Pieter J. Mosterman (2002), "An introduction to multi-paradigm modelling and simulation", Paper read at AI, Simulation and Planning in High Autonomy Systems, at Lisbon, Portugal.

Versweyveld, L. (2006), "Promising but challenging tumour growth simulation experiments at TU Dresden", Virtual Medical Records:722X.

Vesely, FJ. Lattice Gas Cellular Automata. 2001 [cited. Available from http: www.ap.univie.ac.at users ves cp0102 dx node126.html].

Villa, Ferdinando (2001), "Integrating modelling architecture: a declarative framework for multi-paradigm, multi-scale ecological modelling." Ecological Modelling 137 (1):23-42.

Villa, Ferdinando, and Robert Constanza (2000), "Design of multi-paradigm integrating modelling tools for ecological research", Environmental Modelling & Software 15:169-177.

Vlachos, C., R. Gregory, RC. Paton, JR. Saunders, and QH. Wu (2004), "Individual-based modelling of bacterial ecologies and evolution." Comparative and Functional Genomics 5:100-104.

Voitikova, MV. Cellular automaton model for immunology of tumour growth. Comp. Gas (arXiv) 1998 [cited. Available from http: arXiv.org abs comp-gas 9811001].

Walker, DC., J. Southgate, G. Hill, M. Holcombe, DR. Hose, SM. Wood, S. MacNeil, and RH. Smallwood (2004), "The epitheliome: agent-based modelling of the social behaviour of cells." BioSystems 76 (1-3):89-100.

Wang, J., and M. Liu A Formal Model Integration [cited. Available from http: www.dsmforum.org events DSM03 wang.pdf].

Waschek, T., S. Levegrun, M. van Kampen, M. Glesner, R. Engenhart-Cabillic, and W. Schlegel (1997), "Determination of target volumes for three-dimensional radiotherapy of cancer patients with a fuzzy system." Fuzzy Sets and Systems 89:361-370.

Wein, L. Mathematical Modeling of Brain Cancer to Identify Promising Combination Treatments. 1999 [cited. Available from http: virtualtrials.com weinrep2.pdf].

Weiss, G (1999), Multiagent Systems: A modern approach to distributed artificial intelligence. Edited by G Weiss: MIT Press.

Wiley, SH., SY. Shvartsman, and DA. Lauffenburger (2002), "Computational modeling of the EGF-receptor system: a paradigm for systems biology." Trends in Cell Biology 13 (1):43-50.

Winsor, CP. (1932), "The Gompertz curve as a growth curve", Proceedings in the National Academy of Science USA 18:1-7.

Wirtz, FJ. Diffusion Limited Aggregation and its Simulation 2003 [cited. Available from http: www.oche.de ~ecotopia.dla].

Wolfram, S. (1982), "Cellular Automata as Simple Self-Organizing Systems", Caltech preprint CALT-68-938.

Wolfram, S (1988), "Complex Systems Theory", Paper read at Emerging Synthesis in Science: Proceedings of the Founding Workshops of the Santa Fe Institute, at Santa Fe Institute.

Wolfram, S (1994), Cellular Automata. Oxford, UK: Perseus Books Group.

Wolfram, S (2002) Some Historical Notes [cited. Available from http: www.wolframscience.com reference notes 876b].

Wolfram, S (2002), A New Kind of Science: Wolfram Media.

Wolkenhauer, O. (2001), "Systems Biology: The reincarnation of systems theory applied in biology." Briefings in Bioinformatics 2 (3):258-270.

Wolkenhauer, O. (2001), Data Engineering: Fuzzy Mathematics in Systems Theory and Data Analysis. UK: Wiley Inter-Science.

Wolkenhauer, O., KH. Cho, and W. Kolch (2004), "System Biology: Towards an Understanding of the Dynamics of Life", Bioforum Europe 1:50-52.

Wolkenhauer, O., H. Kitano, and KH. Cho (2003), "Systems Biology." IEEE Control Systems Magazine 23 (4):38-48.

Wolkenhauer, O., M. Ullah, W. Kolch, and KH. Cho (2004), "Modelling and Simulation of IntraCellular Dynamics:

Choosing an Appropriate Framework." IEEE Transactions on Nano-Bioscience 3 (3).

Xirasagar, S., S. Gustafson, A. Merrick, KB. Tomer, S. Stasiewicz, D. Chan, KJ. Yost, S. Sumner, N. Xiao, and MD. Waters (2004), "CEBS object model for systems biology data, SysBio-OM." Bioinformatics 20:2004-2015.

Yancopoulos, George D., Samuel Davis, Nicholas W. Gale, John S. Rudge, Stanley J. Wiegand, and Jocelyn Holash (2000), "Vascular-specific growth factors and blood vessel formation", Nature 407:242-248.

Yugi, K., and M. Tomita (2004), "A general computational model of mitochondrial metabolism in a whole organelle scale." Bioinformatics 20 (11):1795-1796.

Zacharaki, E. I., G. S. Stamatakos, K. S. Nikita, and N. K. Uzunoglu (2004), "Simulating growth dynamics and radiation response of avascular tumour spheroids-model validation in the case of an EMT6 Ro multicellular spheroid", Comput Methods Programs Biomed 76 (3):193-206.

Zaider, M., and GN. Minerbo (2000), "Tumour control probability: a formulation applicable to any temporal protocol of dose delivery." Physics in Medicine and Biology 45:279-293.

Zeigler, B., and S. Vahie (1993), "DEVS formalism and methodology: Unity of conception diversity of application", Paper read at Proceedings of the 1993 Winter Simulation Conference.

Zevedei-Oancea, I., and S. Schuster (2003), "Topological analysis of metabolic networks based on Petri net theory." In Silico Biology 3:0029.

Zhang, L, CA Athale, and TS. Deisboeck (2007), "Development of a Three-Dimensional Multiscale Agent-Based Tumour Model: Simulating Gene-Protein Interaction Profiles, Cell Phenotypes & Multicellular Patterns in Brain Cancer", Journal of Theoretical Biology 244 (1):96-107.

Zhu, H., S. Huang, and P. Dhar (2003), "The next step in systems biology: simulating the temporospatial dynamics of molecular network." Bioessays 26:68-72.

Zou, M., and S. D. Conzen (2005), "A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data", Bioinformatics 21 (1):71-79.